aws

Panduan Pengguna

AWS Identity and Access Management

AWS Identity and Access Management: Panduan Pengguna

Table of Contents

Apakah IAM itu?

Follow us on Twitter

AWS Identity and Access Management (IAM) adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya. DenganIAM, Anda dapat mengelola izin yang mengontrol AWS sumber daya yang dapat diakses pengguna. Anda gunakan IAM untuk mengontrol siapa yang diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya. IAMmenyediakan infrastruktur yang diperlukan untuk mengontrol otentikasi dan otorisasi untuk Anda Akun AWS.

Identitas

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut Akun AWS pengguna root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensi pengguna root di IAMPanduan Pengguna.

Gunakan IAM untuk mengatur identitas lain selain pengguna root Anda, seperti administrator, analis, dan pengembang, dan memberi mereka akses ke sumber daya yang mereka butuhkan untuk berhasil dalam tugas mereka.

Manajemen akses

Setelah pengguna diaturIAM, mereka menggunakan kredensi masuk mereka untuk mengautentikasi dengan AWS. Otentikasi disediakan dengan mencocokkan kredensi masuk ke prinsipal (IAMpengguna, pengguna gabungan, IAM peran, atau aplikasi) yang dipercaya oleh Akun AWS. Selanjutnya, permintaan dibuat untuk memberikan akses utama ke sumber daya. Akses diberikan sebagai tanggapan atas permintaan otorisasi jika pengguna telah diberi izin ke sumber daya. Misalnya, saat pertama kali masuk ke konsol dan berada di halaman Beranda konsol, Anda tidak mengakses layanan tertentu. Ketika Anda memilih layanan, permintaan otorisasi dikirim ke layanan tersebut dan terlihat untuk melihat apakah identitas Anda ada dalam daftar pengguna yang berwenang, kebijakan apa yang diberlakukan untuk mengontrol tingkat akses yang diberikan, dan kebijakan lain yang mungkin berlaku. Permintaan otorisasi dapat dilakukan oleh kepala sekolah di dalam Anda Akun AWS atau dari yang lain Akun AWS yang Anda percayai.

Setelah diotorisasi, kepala sekolah dapat mengambil tindakan atau melakukan operasi pada sumber daya di Akun AWS. Misalnya, kepala sekolah dapat meluncurkan yang baru Amazon Elastic Compute Cloud misalnya, memodifikasi keanggotaan IAM grup, atau menghapus Amazon Simple Storage Service ember.

> **Tip**
>
> AWS Pelatihan dan Sertifikasi menyediakan pengantar video 10 menit untukIAM: Pengantar AWS Identity and Access Management.

Ketersediaan layanan

IAM, seperti banyak lainnya AWS Layanan, pada akhirnya konsisten. IAMmencapai ketersediaan tinggi dengan mereplikasi data di beberapa server dalam pusat data Amazon di seluruh dunia. Jika permintaan untuk mengubah beberapa data berhasil, perubahan tersebut akan dilakukan dan disimpan dengan aman. Namun, perubahan harus direplikasiIAM, yang bisa memakan waktu. Perubahan tersebut mencakup membuat atau memperbarui pengguna, kelompok, peran, atau kebijakan. Kami menyarankan agar Anda tidak menyertakan IAM perubahan tersebut dalam jalur kode kritis dan ketersediaan tinggi aplikasi Anda. Sebagai gantinya, buat IAM perubahan dalam inisialisasi atau pengaturan rutin terpisah yang jarang Anda jalankan. Selain itu, pastikan untuk memverifikasi bahwa perubahan telah dibuat merata sebelum alur kerja produksi bergantung padanya. Untuk informasi selengkapnya, lihat Perubahan yang saya buat tidak selalu langsung terlihat.

Informasi biaya layanan

AWS Identity and Access Management (IAM), AWS IAM Identity Center and AWS Security Token Service (AWS STS) adalah fitur Anda AWS Akun yang ditawarkan tanpa biaya tambahan. Anda hanya dikenakan biaya ketika Anda mengakses orang lain AWS layanan menggunakan IAM pengguna Anda atau AWS STS kredensil keamanan sementara.

IAMAnalisis akses eksternal Access Analyzer ditawarkan tanpa biaya tambahan. Namun, Anda akan dikenakan biaya untuk analisis akses yang tidak digunakan dan pemeriksaan kebijakan pelanggan. Untuk daftar lengkap biaya dan harga IAM Access Analyzer, lihat harga IAMAccess Analyzer.

Untuk informasi tentang harga lainnya AWS produk, lihat halaman harga Amazon Web Services.

Integrasi dengan yang lain AWS layanan

IAMterintegrasi dengan banyak AWS layanan. Untuk daftar AWS layanan yang bekerja dengan IAM dan IAM fitur dukungan layanan, lihatAWS layanan yang bekerja dengan IAM.

Untuk informasi lebih lanjut tentang IAM konsep, lihat topik berikut:

Topik

Mengapa saya harus menggunakanIAM?

AWS Identity and Access Management adalah alat yang ampuh untuk mengelola akses ke AWS sumber daya Anda dengan aman. Salah satu manfaat utama menggunakan IAM adalah kemampuan untuk memberikan akses bersama ke AWS akun Anda. Selain itu, IAM memungkinkan Anda untuk menetapkan izin granular, memungkinkan Anda untuk mengontrol dengan tepat tindakan apa yang dapat dilakukan pengguna yang berbeda pada sumber daya tertentu. Tingkat kontrol akses ini sangat penting untuk menjaga keamanan AWS lingkungan Anda. IAMjuga menyediakan beberapa fitur keamanan lainnya. Anda dapat menambahkan otentikasi multi-faktor (MFA) untuk lapisan perlindungan tambahan, dan memanfaatkan federasi identitas untuk mengintegrasikan pengguna dengan mulus dari jaringan perusahaan Anda atau penyedia identitas lainnya. IAMjuga terintegrasi dengan AWS CloudTrail, menyediakan informasi pencatatan dan identitas terperinci untuk mendukung persyaratan audit dan kepatuhan. Dengan memanfaatkan kemampuan ini, Anda dapat membantu memastikan bahwa akses ke AWS sumber daya penting Anda dikontrol dengan ketat dan aman.

Akses bersama ke Akun AWS

Anda dapat memberikan izin kepada orang lain untuk mengelola dan menggunakan sumber daya dalam akun AWS Anda yang tanpa harus membagikan kata sandi atau access key Anda.

Izin granular

Anda dapat memberikan izin yang berbeda kepada orang yang berbeda untuk sumber daya yang berbeda. Misalnya, Anda dapat mengizinkan beberapa pengguna menyelesaikan akses ke Amazon Elastic Compute Cloud (AmazonEC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift, dan layanan lainnya. AWS Untuk pengguna lain, Anda dapat mengizinkan akses hanya-baca ke beberapa bucket Amazon S3 saja, atau izin untuk mengelola beberapa instans EC2 Amazon saja, atau untuk mengakses informasi penagihan Anda tetapi tidak ada yang lain.

Akses aman ke AWS sumber daya untuk aplikasi yang berjalan di Amazon EC2

Anda dapat menggunakan IAM fitur untuk menyediakan kredensil secara aman untuk aplikasi yang berjalan pada instance. EC2 Kredensil ini memberikan izin bagi aplikasi Anda untuk mengakses sumber daya lain. AWS Contohnya termasuk bucket S3 dan tabel DynamoDB.

Otentikasi multi-faktor () MFA

Anda dapat menambahkan autentikasi dua faktor ke akun Anda dan ke pengguna individu untuk keamanan ekstra. Dengan MFA Anda atau pengguna Anda harus memberikan tidak hanya kata sandi atau kunci akses untuk bekerja dengan akun Anda, tetapi juga kode dari perangkat yang dikonfigurasi secara khusus. Jika Anda sudah menggunakan kunci FIDO keamanan dengan layanan lain, dan memiliki konfigurasi yang AWS didukung, Anda dapat menggunakannya WebAuthn untuk MFA keamanan. Untuk informasi selengkapnya, lihat Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

Federasi identitas

Anda dapat mengizinkan pengguna yang sudah memiliki kata sandi di tempat lain—misalnya, di jaringan perusahaan Anda atau dengan penyedia identitas internet—untuk mengakses Anda. Akun AWS Pengguna ini diberikan kredensil sementara yang mematuhi rekomendasi praktik IAM terbaik. Menggunakan federasi identitas meningkatkan keamanan AWS akun Anda.

Informasi identitas untuk jaminan

Jika menggunakan AWS CloudTrail, Anda menerima catatan log yang memuat informasi tentang pihak yang meminta sumber daya di akun Anda. Informasi itu didasarkan pada IAM identitas.

PCIDSSKepatuhan

IAMmendukung pemrosesan, penyimpanan, dan transmisi data kartu kredit oleh pedagang atau penyedia layanan, dan telah divalidasi sebagai sesuai dengan Industri Kartu Pembayaran (PCI) Standar Keamanan Data (DSS). Untuk informasi selengkapnya PCIDSS, termasuk cara meminta salinan Paket AWS PCI Kepatuhan, lihat PCIDSSLevel 1.

Kapan saya menggunakanIAM?

AWS Identity and Access Management adalah layanan infrastruktur inti yang menyediakan dasar untuk kontrol akses berdasarkan identitas di dalamnya AWS. Anda menggunakan IAM setiap kali Anda mengakses AWS akun Anda. Cara Anda menggunakan IAM akan tergantung pada tanggung jawab spesifik dan fungsi pekerjaan dalam organisasi Anda. Pengguna AWS layanan menggunakan IAM untuk mengakses AWS sumber daya yang diperlukan untuk day-to-day pekerjaan mereka, dengan administrator memberikan izin yang sesuai. IAMAdministrator, di sisi lain, bertanggung jawab untuk mengelola IAM identitas dan menulis kebijakan untuk mengontrol akses ke sumber daya. Terlepas dari peran Anda, Anda berinteraksi dengan IAM setiap kali Anda mengautentikasi dan mengotorisasi akses ke AWS sumber daya. Ini bisa melibatkan masuk sebagai IAM pengguna, mengambil IAM peran, atau memanfaatkan federasi identitas untuk akses tanpa batas. Memahami berbagai IAM kemampuan dan kasus penggunaan sangat penting untuk mengelola akses aman ke AWS lingkungan Anda secara efektif. Dalam hal membuat kebijakan dan izin, IAM berikan pendekatan yang fleksibel dan terperinci. Anda dapat menentukan kebijakan kepercayaan untuk mengontrol prinsipal mana yang dapat mengambil peran, selain kebijakan berbasis identitas yang menentukan tindakan dan sumber daya yang dapat diakses pengguna atau peran. Dengan mengonfigurasi IAM kebijakan ini, Anda dapat membantu memastikan bahwa pengguna dan aplikasi memiliki tingkat izin yang sesuai untuk melakukan tugas yang diperlukan.

Saat Anda melakukan fungsi pekerjaan yang berbeda

AWS Identity and Access Management adalah layanan infrastruktur inti yang menyediakan dasar untuk kontrol akses berdasarkan identitas di dalamnya AWS. Anda menggunakan IAM setiap kali Anda mengakses AWS akun Anda.

Cara Anda menggunakan IAM berbeda, tergantung pada pekerjaan yang Anda lakukan AWS.

- Pengguna layanan — Jika Anda menggunakan AWS layanan untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan fitur yang lebih canggih untuk melakukan pekerjaan Anda, Anda mungkin

memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda.

- Administrator layanan — Jika Anda bertanggung jawab atas AWS sumber daya di perusahaan Anda, Anda mungkin memiliki akses penuh keIAM. Tugas Anda adalah menentukan IAM fitur dan sumber daya mana yang harus diakses pengguna layanan Anda. Anda kemudian harus mengirimkan permintaan ke IAM administrator Anda untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasarIAM.
- IAMadministrator — Jika Anda seorang IAM administrator, Anda mengelola IAM identitas dan menulis kebijakan untuk mengelola akses keIAM.

Ketika Anda berwenang untuk mengakses AWS sumber daya

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai IAM pengguna, atau dengan mengambil peranIAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensi yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (Pusat IAM Identitas), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas federasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan IAM peran. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensil Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang menggunakan metode yang disarankan untuk menandatangani permintaan sendiri, lihat Menandatangani AWS API permintaan di Panduan IAM Pengguna.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari lebih lanjut, lihat

Autentikasi multi-faktor di Panduan AWS IAM Identity Center Pengguna dan Menggunakan otentikasi multi-faktor (MFA) AWS di Panduan Pengguna. IAM

Saat Anda masuk sebagai pengguna IAM

IAMPengguna adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, sebaiknya mengandalkan kredensi sementara daripada membuat IAM pengguna yang memiliki kredensi jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensi jangka panjang dengan IAM pengguna, kami sarankan Anda memutar kunci akses. Untuk informasi selengkapnya, lihat Memutar kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensi jangka panjang di IAMPanduan Pengguna.

IAMGrup adalah identitas yang menentukan kumpulan IAM pengguna. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup bernama IAMAdminsdan memberikan izin grup tersebut untuk mengelola sumber dayaIAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari lebih lanjut, lihat Kapan membuat IAM pengguna (bukan peran) di Panduan IAM Pengguna.

Ketika Anda mengambil IAM peran

IAMPeran adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Ini mirip dengan IAM pengguna, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil IAM peran sementara AWS Management Console dengan beralih peran. Anda dapat mengambil peran dengan memanggil AWS CLI atau AWS API operasi atau dengan menggunakan kustomURL. Untuk informasi selengkapnya tentang metode penggunaan peran, lihat Metode untuk mengambil peran dalam Panduan IAM Pengguna.

IAMperan dengan kredensi sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk

informasi tentang peran untuk federasi, lihat Membuat peran untuk Penyedia Identitas pihak ketiga di Panduan IAM Pengguna. Jika Anda menggunakan Pusat IAM Identitas, Anda mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah diautentikasi, Pusat IAM Identitas mengkorelasikan izin yang disetel ke peran. IAM Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .

- Izin IAM pengguna sementara — IAM Pengguna atau peran dapat mengambil IAM peran untuk sementara mengambil izin yang berbeda untuk tugas tertentu.
- Akses lintas akun — Anda dapat menggunakan IAM peran untuk memungkinkan seseorang (prinsipal tepercaya) di akun lain mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM Panduan Pengguna. IAM
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan IAM pengguna atau peran untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FASmenggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. FASPermintaan hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan saat membuat FAS permintaan, lihat Meneruskan sesi akses.
  - Peran layanan — Peran layanan adalah IAMperan yang diasumsikan layanan untuk melakukan tindakan atas nama Anda. IAMAdministrator dapat membuat, memodifikasi, dan menghapus peran layanan dari dalamIAM. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke Layanan AWS dalam IAMPanduan Pengguna.
  - Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke peran layanan. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. IAMAdministrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.

- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan IAM peran untuk mengelola kredensil sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS API meminta. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensi sementara. Untuk informasi selengkapnya, lihat Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di IAMPanduan Pengguna.

Untuk mempelajari apakah akan menggunakan IAM peran atau IAM pengguna, lihat Kapan membuat IAM peran (bukan pengguna) di Panduan IAM Pengguna.

Saat Anda membuat kebijakan dan izin

Anda memberikan izin kepada pengguna dengan membuat kebijakan, yaitu dokumen yang mencantumkan tindakan yang dapat dilakukan pengguna dan sumber daya yang dapat memengaruhi tindakan tersebut. Setiap tindakan atau sumber daya yang tidak diizinkan secara eksplisit ditolak secara default. Kebijakan dapat dibuat dan dilampirkan ke prinsipal (pengguna, kelompok pengguna, peran yang diasumsikan oleh pengguna, dan sumber daya).

Anda dapat menggunakan kebijakan ini dengan IAM peran:

- Kebijakan kepercayaan — Mendefinisikan prinsipal mana yang dapat mengambil peran, dan dalam kondisi apa. Kebijakan kepercayaan adalah jenis kebijakan berbasis sumber daya khusus untuk peran. IAM Peran hanya dapat memiliki satu kebijakan kepercayaan.
- Kebijakan berbasis identitas (sebaris dan terkelola) — Kebijakan ini menentukan izin yang dapat dilakukan oleh pengguna peran (atau ditolak untuk melakukan), dan sumber daya mana.

Gunakan Contoh kebijakan berbasis identitas IAM untuk membantu Anda menentukan izin untuk IAM identitas Anda. Setelah menemukan kebijakan yang diperlukan, pilih lihat kebijakan untuk melihat kebijakan tersebut. JSON Anda dapat menggunakan dokumen JSON kebijakan sebagai templat untuk kebijakan Anda sendiri.

> **Note**
>
> Jika Anda menggunakan Pusat IAM Identitas untuk mengelola pengguna, Anda menetapkan set izin di Pusat IAM Identitas alih-alih melampirkan kebijakan izin ke prinsipal. Saat Anda

menetapkan izin yang disetel ke grup atau pengguna di Pusat Identitas AWS IAM, Pusat IAM Identitas akan membuat IAM peran yang sesuai di setiap akun, dan melampirkan kebijakan yang ditentukan dalam izin yang disetel ke peran tersebut. IAMPusat Identitas mengelola peran, dan memungkinkan pengguna resmi yang telah Anda tetapkan untuk mengambil peran tersebut. Jika Anda mengubah set izin, Pusat IAM Identitas memastikan bahwa IAM kebijakan dan peran terkait diperbarui sesuai dengan itu.
Untuk informasi selengkapnya tentang Pusat IAM Identitas, lihat Apa itu Pusat IAM Identitas? dalam AWS IAM Identity Center User Guide.

Bagaimana cara saya mengelolaIAM?

Mengelola AWS Identity and Access Management dalam suatu AWS lingkungan melibatkan pemanfaatan berbagai alat dan antarmuka. Metode yang paling umum adalah melalui AWS Management Console, antarmuka berbasis web yang memungkinkan Anda melakukan berbagai tugas IAM administratif, mulai dari membuat pengguna dan peran hingga mengonfigurasi izin.

Untuk pengguna yang lebih nyaman dengan antarmuka baris perintah, AWS menyediakan dua set alat baris perintah - AWS Command Line Interface dan. AWS Tools for Windows PowerShell Ini memungkinkan Anda untuk mengeluarkan perintah IAM terkait langsung dari terminal, seringkali lebih efisien daripada menavigasi konsol. Selain itu, AWS CloudShell memungkinkan Anda untuk menjalankan CLI atau SDK perintah langsung dari browser web Anda, menggunakan izin yang terkait dengan login konsol Anda.

Di luar konsol dan baris perintah, AWS menawarkan Software Development Kits (SDKs) untuk berbagai bahasa pemrograman, memungkinkan Anda untuk mengintegrasikan fungsionalitas IAM manajemen langsung ke aplikasi Anda. Atau, Anda dapat mengakses IAM secara terprogram menggunakan IAM QueryAPI, yang memungkinkan Anda untuk mengeluarkan HTTPS permintaan langsung ke layanan. Memanfaatkan pendekatan manajemen yang berbeda ini memberi Anda fleksibilitas untuk dimasukkan IAM ke dalam alur kerja dan proses yang ada.

Gunakan AWS Management Console

AWS Management Console adalah aplikasi web yang terdiri dan mengacu pada koleksi luas konsol layanan untuk mengelola AWS sumber daya. Saat pertama kali masuk, Anda akan melihat halaman beranda konsol. Halaman beranda menyediakan akses ke setiap konsol layanan dan menawarkan satu tempat untuk mengakses informasi untuk melakukan tugas AWS terkait Anda. Layanan dan aplikasi mana yang tersedia untuk Anda setelah masuk ke konsol bergantung pada AWS sumber

daya mana yang dapat Anda akses. Anda dapat diberikan izin untuk sumber daya baik melalui asumsi peran, menjadi anggota grup yang telah diberikan izin, atau secara eksplisit diberikan izin. Untuk AWS akun yang berdiri sendiri, pengguna root atau IAM administrator mengonfigurasi akses ke sumber daya. Untuk AWS Organizations, akun manajemen atau administrator yang didelegasikan mengonfigurasi akses ke sumber daya.

Jika Anda berencana untuk memiliki orang yang menggunakan Konsol AWS Manajemen untuk mengelola AWS sumber daya, sebaiknya mengonfigurasi pengguna dengan kredensil sementara sebagai praktik terbaik keamanan. IAMpengguna yang telah mengambil peran, pengguna federasi, dan pengguna di IAM Identity Center memiliki kredensi sementara, sedangkan IAM pengguna dan pengguna root memiliki kredensi jangka panjang. Kredensi pengguna root menyediakan akses penuh ke Akun AWS, sementara pengguna lain memiliki kredensil yang menyediakan akses ke sumber daya yang diberikan oleh kebijakan. IAM

Pengalaman masuk berbeda untuk berbagai jenis AWS Management Console pengguna.

- IAMpengguna dan pengguna root masuk dari AWS login utama URL ()https://signin.aws.amazon.com. Setelah mereka masuk, mereka memiliki akses ke sumber daya di akun yang telah diberikan izin kepada mereka.

  Untuk masuk sebagai pengguna root, Anda harus memiliki alamat email dan kata sandi pengguna root.

  Untuk masuk sebagai IAM pengguna, Anda harus memiliki Akun AWS nomor atau alias, nama IAM pengguna, dan kata sandi IAM pengguna.

  Kami menyarankan Anda membatasi IAM pengguna di akun Anda untuk situasi tertentu yang memerlukan kredensi jangka panjang, seperti untuk akses darurat, dan bahwa Anda menggunakan pengguna root hanya untuk tugas-tugas yang memerlukan kredensi pengguna root.

  Untuk kenyamanan, halaman AWS masuk menggunakan cookie browser untuk mengingat nama IAM pengguna dan informasi akun. Lain kali pengguna pergi ke halaman mana pun di AWS Management Console, konsol menggunakan cookie untuk mengarahkan pengguna ke halaman masuk akun.

  Keluar dari konsol saat Anda menyelesaikan sesi untuk mencegah penggunaan kembali login sebelumnya.
- IAMPengguna Pusat Identitas masuk menggunakan portal AWS akses khusus yang unik untuk organisasi mereka. Setelah mereka masuk, mereka dapat memilih akun atau aplikasi mana yang

akan diakses. Jika mereka memilih untuk mengakses akun, mereka memilih set izin yang ingin mereka gunakan untuk sesi manajemen.

- Pengguna gabungan yang dikelola di penyedia identitas eksternal yang ditautkan ke Akun AWS login menggunakan portal akses perusahaan khusus. Sumber AWS daya yang tersedia untuk pengguna federasi tergantung pada kebijakan yang dipilih oleh organisasi mereka.

> **Note**
>
> Untuk memberikan tingkat keamanan tambahan, pengguna root, pengguna, dan IAM pengguna di Pusat IAM Identitas dapat memiliki otentikasi multi-faktor (MFA) yang diverifikasi AWS sebelum memberikan akses ke sumber daya. AWS Ketika MFA diaktifkan, Anda juga harus memiliki akses ke MFA perangkat untuk masuk.

Untuk mempelajari lebih lanjut tentang cara pengguna yang berbeda masuk ke konsol manajemen, lihat Masuk ke Konsol AWS Manajemen di Panduan Pengguna AWS Masuk.

AWS Alat Baris Perintah

Anda dapat menggunakan alat baris AWS perintah untuk mengeluarkan perintah di baris perintah sistem Anda untuk melakukan IAM dan AWS tugas. Menggunakan baris perintah dapat lebih cepat dan lebih nyaman dibandingkan konsol. Alat baris perintah juga berguna jika Anda ingin membangun skrip yang melakukan AWS tugas.

AWS menyediakan dua set alat baris perintah: AWS Command Line Interface(AWS CLI) dan AWS Tools for Windows PowerShell. Untuk informasi tentang menginstal dan menggunakan AWS CLI, lihat Panduan AWS Command Line Interface Pengguna. Untuk informasi tentang menginstal dan menggunakan Alat untuk Windows PowerShell, lihat Panduan AWS Tools for Windows PowerShell Pengguna.

Setelah masuk ke konsol, Anda dapat menggunakan AWS CloudShell dari browser Anda untuk menjalankan CLI atau SDK perintah. Izin untuk mengakses AWS sumber daya didasarkan pada kredenal yang Anda gunakan untuk masuk ke konsol. Tergantung pada pengalaman Anda, Anda mungkin menemukan metode yang lebih efisien CLI untuk mengelola Anda Akun AWS. Untuk informasi selengkapnya, silakan lihat Gunakan AWS CloudShell untuk bekerja dengan AWS Identity and Access Management

AWS Antarmuka Baris Perintah (CLI) dan Kit Pengembangan Perangkat Lunak () SDKs

IAMPusat Identitas dan IAM pengguna menggunakan metode yang berbeda untuk mengautentikasi kredensialnya ketika mereka mengautentikasi melalui CLI atau antarmuka aplikasi () APIs di yang terkait. SDKs

Kredensial dan pengaturan konfigurasi terletak di beberapa tempat, seperti variabel sistem atau lingkungan pengguna, file AWS konfigurasi lokal, atau secara eksplisit dinyatakan pada baris perintah sebagai parameter. Lokasi tertentu lebih diutamakan daripada yang lain.

Baik Pusat IAM Identitas dan IAM menyediakan kunci akses yang dapat digunakan dengan CLI atauSDK. IAMKunci akses Pusat Identitas adalah kredensil sementara yang dapat disegarkan secara otomatis dan direkomendasikan melalui kunci akses jangka panjang yang terkait dengan pengguna. IAM

Untuk mengelola Anda Akun AWS menggunakan CLI atau SDK Anda dapat menggunakan AWS CloudShell dari browser Anda. Jika Anda menggunakan CloudShell untuk menjalankan CLI atau SDK perintah, Anda harus terlebih dahulu masuk ke konsol. Izin untuk mengakses AWS sumber daya didasarkan pada kredenal yang Anda gunakan untuk masuk ke konsol. Tergantung pada pengalaman Anda, Anda mungkin menemukan metode yang lebih efisien CLI untuk mengelola Anda Akun AWS.

Untuk pengembangan aplikasi, Anda dapat mengunduh CLI atau SDK ke komputer Anda dan masuk dari prompt perintah atau jendela Docker. Dalam skenario ini, Anda mengonfigurasi otentikasi dan mengakses kredensil sebagai bagian dari CLI skrip atau aplikasi. SDK Anda dapat mengonfigurasi akses terprogram ke sumber daya dengan cara yang berbeda, tergantung pada lingkungan dan akses yang tersedia untuk Anda.

- Opsi yang disarankan untuk mengautentikasi kode lokal dengan AWS layanan adalah Pusat IAM Identitas dan IAM Peran Di Mana Saja
- Opsi yang disarankan untuk mengautentikasi kode yang berjalan dalam AWS lingkungan adalah dengan menggunakan IAM peran atau menggunakan kredensil Pusat IAM Identitas.

Saat masuk menggunakan portal AWS akses, Anda bisa mendapatkan kredensi jangka pendek dari halaman awal tempat Anda memilih set izin. Kredensi ini memiliki durasi yang ditentukan dan tidak disegarkan secara otomatis. Jika Anda ingin menggunakan kredensil ini, setelah masuk ke AWS portal, pilih Akun AWS dan kemudian pilih set izin. Pilih Baris perintah atau akses terprogram untuk

melihat opsi yang dapat Anda gunakan untuk mengakses AWS sumber daya secara terprogram atau dari. CLI Untuk informasi selengkapnya tentang metode ini, lihat Mendapatkan dan menyegarkan kredensil sementara di Panduan Pengguna Pusat IAM Identitas. Kredensi ini sering digunakan selama pengembangan aplikasi untuk menguji kode dengan cepat.

Sebaiknya gunakan IAM kredensil Pusat Identitas yang secara otomatis menyegarkan saat mengotomatiskan akses ke sumber daya Anda. AWS Jika Anda telah mengonfigurasi pengguna dan set izin di Pusat IAM Identitas, Anda menggunakan `aws configure sso` perintah untuk menggunakan wizard baris perintah yang akan membantu Anda mengidentifikasi kredensil yang tersedia untuk Anda dan menyimpannya di profil. Untuk informasi selengkapnya tentang mengonfigurasi profil Anda, lihat Mengonfigurasi profil Anda dengan `aws configure sso` wizard di Panduan Pengguna Antarmuka Baris AWS Perintah untuk Versi 2.

> **Note**
>
> Banyak contoh aplikasi menggunakan kunci akses jangka panjang yang terkait dengan IAM pengguna atau pengguna root. Anda hanya boleh menggunakan kredensil jangka panjang dalam lingkungan kotak pasir sebagai bagian dari latihan pembelajaran. Tinjau alternatif untuk kunci akses jangka panjang dan rencanakan untuk mentransisikan kode Anda untuk menggunakan kredensil alternatif, seperti kredensi atau IAM peran Pusat IAM Identitas, sesegera mungkin. Setelah mentransisikan kode Anda, hapus kunci akses.

Untuk mempelajari lebih lanjut tentang mengonfigurasiCLI, lihat Menginstal atau memperbarui versi terbaru AWS CLI dalam Panduan Pengguna Antarmuka Baris AWS Perintah untuk Versi 2 dan Autentikasi dan akses kredensional di Panduan Pengguna Antarmuka Baris AWS Perintah

Untuk mempelajari lebih lanjut tentang mengonfigurasiSDK, lihat otentikasi Pusat IAM Identitas di Panduan Referensi Alat AWS SDKs dan IAMPeran Di Mana Saja di Panduan Referensi Alat AWS SDKs dan Alat.

Gunakan AWS SDKs

AWS menyediakan SDKs (perangkat pengembangan perangkat lunak) yang terdiri dari perpustakaan dan kode sampel untuk berbagai bahasa dan platform pemrograman (Java, Python, Ruby,. NET, iOS, Android, dll.). SDKsMenyediakan cara yang nyaman untuk membuat akses terprogram ke IAM dan AWS. Misalnya, SDKs mengurus tugas-tugas seperti menandatangani permintaan secara kriptografis, mengelola kesalahan, dan mencoba ulang permintaan secara otomatis. Untuk informasi

tentang AWS SDKs, termasuk cara mengunduh dan menginstalnya, lihat halaman Alat untuk Amazon Web Services.

Gunakan IAM Query API

Anda dapat mengakses IAM dan AWS secara terprogram menggunakan IAM KueriAPI, yang memungkinkan Anda mengeluarkan HTTPS permintaan langsung ke layanan. Saat Anda menggunakan KueriAPI, Anda harus menyertakan kode untuk menandatangani permintaan secara digital menggunakan kredensil Anda. Untuk informasi lebih lanjut, lihat Memanggil permintaan HTTP kueri IAM API menggunakan dan IAMAPIReferensi.

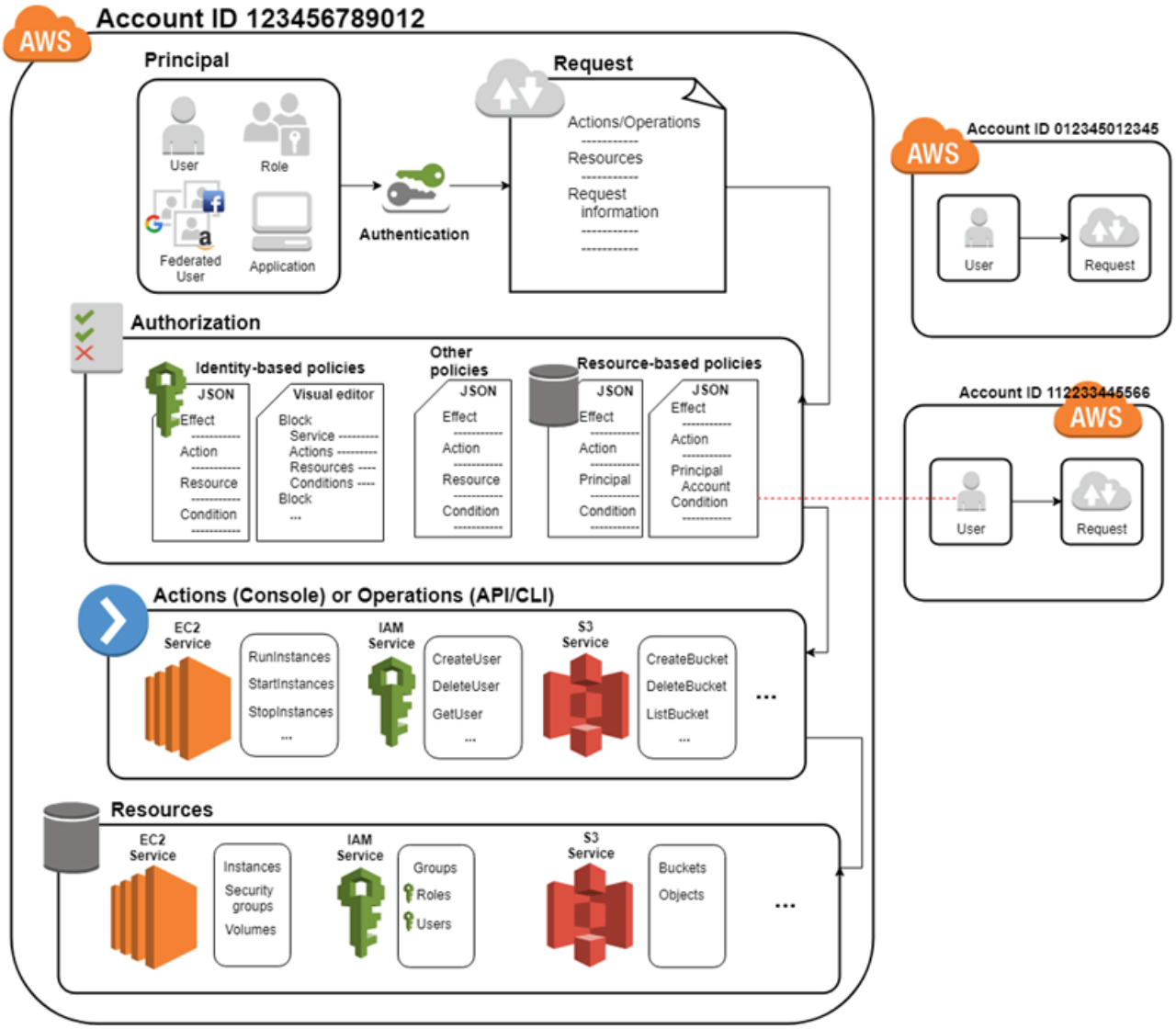
Cara kerja IAM

AWS Identity and Access Management menyediakan infrastruktur yang diperlukan untuk mengontrol otentikasi dan otorisasi untuk Anda. Akun AWS

Pertama, pengguna manusia atau aplikasi menggunakan kredensyal masuk mereka untuk mengautentikasi. AWS IAMmencocokkan kredensi masuk dengan prinsipal (IAMpengguna, pengguna gabungan, IAM peran, atau aplikasi) yang dipercaya oleh Akun AWS dan mengautentikasi izin untuk mengakses. AWS

Selanjutnya, IAM buat permintaan untuk memberikan akses utama ke sumber daya. IAMmemberikan atau menolak akses sebagai tanggapan atas permintaan otorisasi. Misalnya, saat pertama kali masuk ke konsol dan berada di halaman Beranda konsol, Anda tidak mengakses layanan tertentu. Ketika Anda memilih layanan, Anda mengirim permintaan otorisasi IAM untuk layanan tersebut. IAMmemverifikasi bahwa identitas Anda ada dalam daftar pengguna yang berwenang, menentukan kebijakan apa yang mengontrol tingkat akses yang diberikan, dan mengevaluasi kebijakan lain yang mungkin berlaku. Prinsipal di dalam Anda Akun AWS atau dari orang lain Akun AWS yang Anda percayai dapat membuat permintaan otorisasi.

Setelah diotorisasi, kepala sekolah dapat melakukan tindakan atau operasi pada sumber daya di Anda Akun AWS. Misalnya, prinsipal dapat meluncurkan Amazon Elastic Compute Cloud instance baru, memodifikasi keanggotaan IAM grup, atau menghapus Amazon Simple Storage Service bucket. Diagram berikut menggambarkan proses ini melalui IAM infrastruktur:



Komponen permintaan

Ketika seorang kepala sekolah mencoba menggunakan AWS Management Console, the AWS API, atau AWS CLI, prinsipal tersebut mengirimkan permintaan ke AWS. Permintaan tersebut mencakup informasi berikut:

- Tindakan atau operasi — Tindakan atau operasi yang ingin dilakukan oleh prinsipal, seperti tindakan dalam AWS Management Console, atau operasi di AWS CLI atau AWS API.
- Resources — Objek AWS sumber daya di mana prinsipal meminta untuk melakukan tindakan atau operasi.
- Penanggung jawab – Orang atau aplikasi yang menggunakan entitas (pengguna atau peran) untuk mengirim permintaan. Informasi tentang kepala sekolah termasuk kebijakan izin.

- Data lingkungan — Informasi tentang alamat IP, agen pengguna, status yang SSL diaktifkan, dan stempel waktu.
- Data sumber daya — Data yang terkait dengan sumber daya yang diminta, seperti nama tabel DynamoDB atau tag pada instance Amazon. EC2

AWS mengumpulkan informasi permintaan ke dalam konteks permintaan, yang IAM mengevaluasi untuk mengotorisasi permintaan.

Bagaimana prinsipal diautentikasi

Seorang kepala sekolah masuk untuk AWS menggunakan kredensialnya yang IAM mengautentikasi untuk mengizinkan kepala sekolah mengirim permintaan. AWS Beberapa layanan, seperti Amazon S3 dan AWS STS, memungkinkan permintaan khusus dari pengguna anonim. Namun, mereka adalah pengecualian dari aturan tersebut. Setiap jenis pengguna melewati otentikasi.

- Pengguna root - Kredensyal masuk Anda yang digunakan untuk otentikasi adalah alamat email yang Anda gunakan untuk membuat Akun AWS dan kata sandi yang Anda tentukan pada saat itu.
- Pengguna federasi — Penyedia identitas Anda mengautentikasi Anda dan meneruskan kredensyal Anda ke AWS, Anda tidak harus langsung masuk. AWS Baik Pusat IAM Identitas dan IAM mendukung pengguna federasi.
- Pengguna di Direktori Pusat Identitas AWS IAM (tidak terfederasi) - Pengguna yang dibuat langsung di direktori default Pusat IAM Identitas masuk menggunakan portal AWS akses dan memberikan nama pengguna dan kata sandi Anda.
- IAMpengguna — Anda masuk dengan memberikan ID akun atau alias, nama pengguna, dan kata sandi Anda. Untuk mengautentikasi beban kerja dari API atau AWS CLI, Anda dapat menggunakan kredensyal sementara dengan mengasumsikan peran atau Anda mungkin menggunakan kredensyal jangka panjang dengan menyediakan kunci akses dan kunci rahasia Anda.

  Untuk mempelajari lebih lanjut tentang IAM entitas, lihat IAMpengguna danIAMperan.

AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) dengan semua pengguna untuk meningkatkan keamanan akun Anda. Untuk mempelajari lebih lanjut tentang MFA, lihat AWS Otentikasi multi-faktor di IAM.

Dasar-dasar kebijakan otorisasi dan izin

Otorisasi mengacu pada prinsipal yang memiliki izin yang diperlukan untuk menyelesaikan permintaan mereka. Selama otorisasi, IAM mengidentifikasi kebijakan yang berlaku untuk permintaan menggunakan nilai dari konteks permintaan. Kemudian menggunakan kebijakan untuk menentukan apakah akan mengizinkan atau menolak permintaan. IAMmenyimpan sebagian besar kebijakan izin sebagai JSONdokumen yang menentukan izin untuk entitas utama.

Ada beberapa jenis kebijakan yang dapat memengaruhi permintaan otorisasi. Untuk memberi pengguna izin untuk mengakses AWS sumber daya di akun, Anda dapat menggunakan kebijakan berbasis identitas. Kebijakan berbasis sumber daya dapat memberikan akses lintas akun. Untuk membuat permintaan di akun yang berbeda, kebijakan di akun lain harus memungkinkan Anda mengakses sumber daya dan IAM entitas yang Anda gunakan untuk membuat permintaan harus memiliki kebijakan berbasis identitas yang memungkinkan permintaan tersebut.

IAMmemeriksa setiap kebijakan yang berlaku untuk konteks permintaan Anda. IAMevaluasi kebijakan menggunakan penolakan eksplisit, yang berarti bahwa jika kebijakan izin tunggal menyertakan tindakan yang ditolak, IAM menolak seluruh permintaan dan berhenti mengevaluasi. Karena permintaan ditolak secara default, kebijakan izin yang berlaku harus mengizinkan setiap bagian dari permintaan Anda IAM untuk mengotorisasi permintaan Anda. Logika evaluasi untuk permintaan dalam satu akun mengikuti aturan dasar ini:

- Secara default, semua permintaan ditolak. (Secara umum, permintaan yang dibuat menggunakan kredensial Pengguna root akun AWS untuk sumber daya di akun selalu diperbolehkan.)
- Izin eksplisit dalam kebijakan izin apa pun (berbasis identitas atau berbasis sumber daya) menggantikan pengaturan bawaan ini.
- Keberadaan OrganizationsSCP, batas IAM izin, atau kebijakan sesi mengesampingkan izin. Jika ada satu atau lebih jenis kebijakan ini, maka semuanya harus mengizinkan permintaan tersebut. Kalau tidak, itu secara implisit ditolak.
- Penolakan eksplisit dalam kebijakan apa pun mengesampingkan izin apa pun dalam kebijakan apa pun.

Untuk mempelajari selengkapnya, lihat Logika evaluasi kebijakan.

Setelah IAM mengotentikasi dan mengotorisasi kepala sekolah, IAM menyetujui tindakan atau operasi dalam permintaan mereka dengan mengevaluasi kebijakan izin yang berlaku untuk kepala sekolah. Setiap AWS layanan mendefinisikan tindakan (operasi) yang mereka dukung,

dan menyertakan hal-hal yang dapat Anda lakukan untuk sumber daya, seperti melihat, membuat, mengedit, dan menghapus sumber daya tersebut. Kebijakan izin yang berlaku untuk kepala sekolah harus mencakup tindakan yang diperlukan untuk melakukan operasi. Untuk mempelajari selengkapnya tentang cara IAM mengevaluasi kebijakan izin, lihatthe section called “Logika evaluasi kebijakan”.

Layanan mendefinisikan serangkaian tindakan yang dapat dilakukan oleh prinsipal pada setiap sumber daya. Saat membuat kebijakan izin, pastikan untuk menyertakan tindakan yang Anda ingin pengguna dapat lakukan. Misalnya, IAM mendukung lebih dari 40 tindakan untuk sumber daya pengguna, termasuk tindakan dasar berikut:

- `CreateUser`
- `DeleteUser`
- `GetUser`
- `UpdateUser`

Selain itu, Anda dapat menentukan kondisi dalam kebijakan izin yang menyediakan akses ke sumber daya saat permintaan memenuhi ketentuan yang ditentukan. Misalnya, Anda mungkin ingin pernyataan kebijakan berlaku setelah tanggal tertentu atau untuk mengontrol akses ketika nilai tertentu muncul diAPI. Untuk menentukan kondisi, Anda menggunakan ??? `Condition` elemen pernyataan kebijakan.

Setelah IAM menyetujui operasi dalam permintaan, kepala sekolah dapat bekerja dengan sumber daya terkait di akun Anda. Sumber daya adalah objek yang ada di dalam layanan. Contohnya termasuk EC2 instans Amazon, IAM pengguna, dan bucket Amazon S3. Jika prinsipal membuat permintaan untuk melakukan tindakan pada sumber daya yang tidak disertakan dalam kebijakan izin, layanan menolak permintaan tersebut. Misalnya, jika Anda memiliki izin untuk menghapus IAM peran tetapi meminta untuk menghapus IAM grup, permintaan gagal jika Anda tidak memiliki izin untuk menghapus IAM grup. Untuk mempelajari selengkapnya tentang tindakan, sumber daya, dan kunci kondisi yang didukung berbagai AWS layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Bandingkan IAM identitas dan kredensialnya

Identitas yang dikelola AWS Identity and Access Management adalah IAM pengguna, IAM peran, dan IAM grup. Identitas ini merupakan tambahan dari pengguna root Anda yang AWS dibuat bersama dengan Anda Akun AWS.

Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari, bahkan tugas administratif. Sebagai gantinya, berikan pengguna tambahan dan beri mereka izin yang diperlukan untuk melakukan tugas yang diperlukan. Anda dapat menambahkan pengguna baik dengan menambahkan orang ke direktori Pusat IAM Identitas Anda, menggabungkan penyedia identitas eksternal dengan Pusat IAM Identitas atau IAM atau membuat pengguna dengan hak istimewa IAM paling sedikit.

Setelah menyiapkan pengguna, Anda dapat memberikan akses ke orang tertentu dan memberi mereka izin untuk mengakses sumber daya. Akun AWS

Sebagai praktik terbaik, AWS merekomendasikan agar Anda meminta pengguna manusia untuk mengambil IAM peran untuk mengakses AWS sehingga mereka menggunakan kredensi sementara. Jika Anda mengelola identitas di direktori Pusat IAM Identitas atau menggunakan federasi dengan penyedia identitas, Anda mengikuti praktik terbaik.

Ketentuan

Istilah-istilah ini biasanya digunakan saat bekerja dengan IAM identitas:

IAMSumber Daya

IAMLayanan menyimpan sumber daya ini. Anda dapat menambahkan, mengedit, dan menghapusnya dari IAM konsol.

- IAMpengguna
- IAMkelompok
- IAMperan
- Kebijakan izin
- Objek penyedia identitas

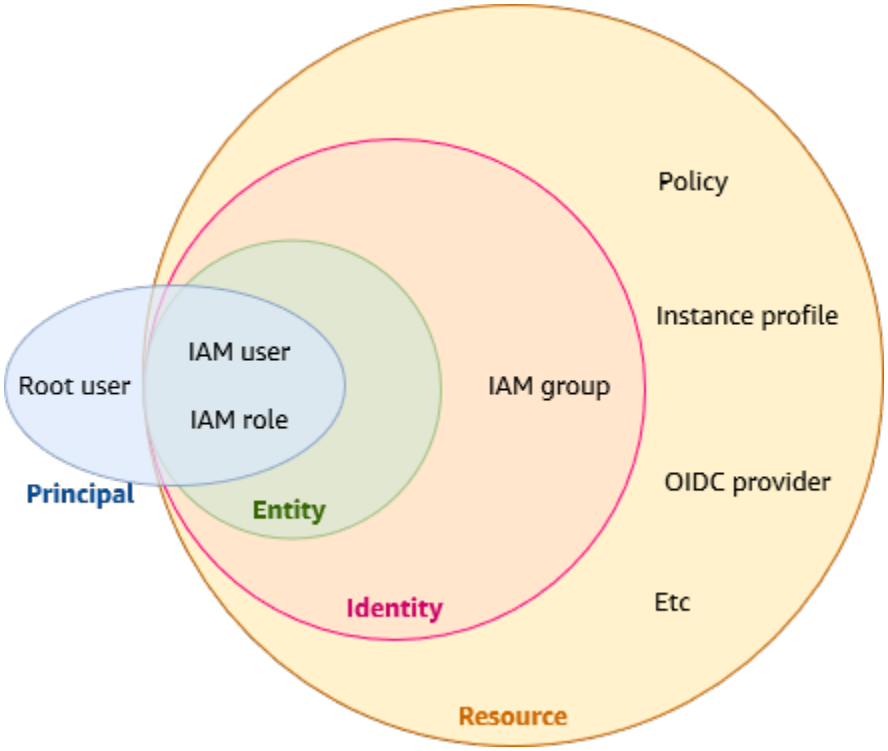
IAMEntitas

IAMsumber daya yang AWS digunakan untuk otentikasi. Tentukan entitas sebagai Principal dalam kebijakan berbasis sumber daya.

- IAMpengguna
- IAMperan

IAMIdentitas

IAMSumber daya yang diizinkan dalam kebijakan untuk melakukan tindakan dan mengakses sumber daya. Identitas termasuk IAM pengguna, IAM grup, dan IAM peran.



Kepala Sekolah

IAMPengguna Pengguna root akun AWS, atau IAM peran yang dapat membuat permintaan untuk tindakan atau operasi pada AWS sumber daya. Prinsipal termasuk pengguna manusia, beban kerja, pengguna federasi dan peran yang diasumsikan. Setelah otentikasi, IAM memberikan kredensi permanen atau sementara kepada prinsipal untuk mengajukan permintaan AWS, tergantung pada jenis prinsipnya.

Pengguna manusia juga dikenal sebagai identitas manusia, seperti orang, administrator, pengembang, operator, dan konsumen aplikasi Anda.

Beban kerja adalah kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi, proses, alat operasional, dan komponen lainnya.

Pengguna federasi adalah pengguna yang identitas dan kredensialnya dikelola oleh penyedia identitas lain, seperti Active Directory, Okta, atau Microsoft Entra.

IAMperan adalah IAM identitas yang dapat Anda buat di akun Anda yang memiliki izin khusus yang menentukan apa yang dapat dan tidak dapat dilakukan oleh identitas tersebut. Namun, alih-alih secara unik terkait dengan satu orang, peran dimaksudkan untuk menjadi dapat diambil oleh siapa pun yang membutuhkannya.

IAMmemberikan IAM kredensi sementara dan IAM peran jangka panjang kepada pengguna dan pengguna root. Pengguna federasi dan pengguna di Pusat Identitas AWS IAM mengambil IAM peran saat mereka masuk AWS, yang memberi mereka kredensi sementara. Sebagai praktik terbaik, kami menyarankan Anda meminta pengguna manusia dan beban kerja untuk mengakses AWS sumber daya menggunakan kredensi sementara.

Perbedaan antara IAM pengguna dan pengguna di Pusat IAM Identitas

IAMpengguna bukan akun terpisah; mereka adalah pengguna individu dalam akun Anda. Setiap pengguna memiliki kata sandi mereka sendiri untuk akses ke file AWS Management Console. Anda juga dapat membuat access key individu untuk setiap pengguna sehingga pengguna dapat membuat permintaan terprogram untuk bekerja dengan sumber daya di akun Anda.

IAMpengguna dan kunci akses mereka memiliki kredensi jangka panjang untuk sumber daya Anda AWS . Penggunaan utama bagi IAM pengguna adalah untuk memberikan beban kerja yang tidak dapat menggunakan IAM peran kemampuan untuk membuat permintaan terprogram ke AWS layanan menggunakan atau. API CLI

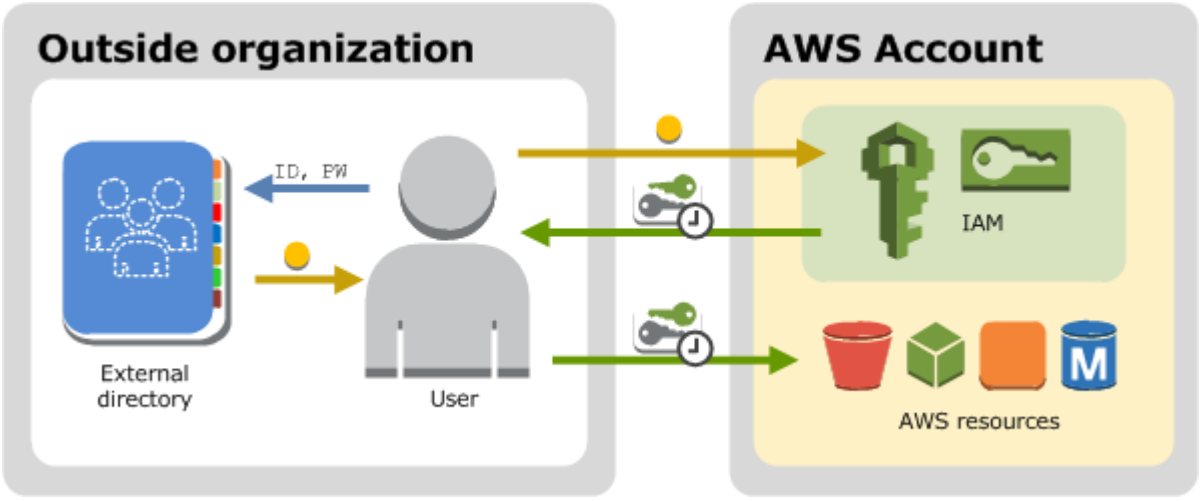
> **Note**
>
> Untuk skenario di mana Anda memerlukan IAM pengguna dengan akses terprogram dan kredensi jangka panjang, kami sarankan Anda memperbarui kunci akses bila diperlukan. Untuk informasi selengkapnya, lihat Perbarui kunci akses.

Identitas tenaga kerja (orang pengguna di Pusat Identitas AWS IAM) memiliki kebutuhan izin yang berbeda tergantung pada peran yang mereka lakukan dan dapat bekerja di Akun AWS berbagai organisasi. Jika Anda memiliki kasus penggunaan yang memerlukan kunci akses, Anda dapat mendukung kasus penggunaan tersebut pengguna di Pusat Identitas AWS IAM. Orang yang masuk melalui portal AWS akses dapat memperoleh kunci akses dengan kredensi jangka pendek ke sumber daya Anda. AWS Untuk manajemen akses terpusat, kami menyarankan Anda menggunakan AWS IAM Identity Center (Pusat IAM Identitas) untuk mengelola akses ke akun Anda dan izin dalam akun tersebut. IAMPusat Identitas secara otomatis dikonfigurasi dengan direktori Pusat Identitas sebagai sumber identitas default tempat Anda dapat menambahkan orang dan grup, dan menetapkan tingkat akses mereka ke AWS sumber daya Anda. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center dalam Panduan Pengguna AWS IAM Identity Center .

Perbedaan utama antara kedua jenis pengguna ini adalah bahwa pengguna di Pusat IAM Identitas secara otomatis mengambil IAM peran ketika mereka masuk AWS sebelum mereka mengakses konsol manajemen atau AWS sumber daya. IAMperan memberikan kredensi sementara setiap kali pengguna masuk. AWS Agar IAM pengguna dapat masuk menggunakan IAM peran, mereka harus memiliki izin untuk mengambil dan beralih peran dan mereka harus secara eksplisit memilih untuk beralih ke peran yang ingin mereka ambil setelah mengakses akun. AWS

Pengguna federasi dari sumber identitas yang ada

Jika pengguna di organisasi Anda sudah diautentikasi saat mereka masuk ke jaringan perusahaan, Anda tidak perlu membuat IAM pengguna atau pengguna terpisah di Pusat IAM Identitas untuk mereka. Sebagai gantinya, Anda dapat menggabungkan identitas pengguna tersebut ke dalam AWS menggunakan salah satu ataulAM. AWS IAM Identity Center Pengguna federasi mengambil IAM peran yang memberi mereka izin untuk mengakses sumber daya tertentu. Untuk informasi lebih lanjut tentang peran, lihat Istilah dan konsep peran.



Federasi berguna dalam kasus ini:

- Pengguna Anda sudah ada di direktori perusahaan.

  Jika direktori perusahaan Anda kompatibel dengan Security Assertion Markup Language 2.0 (SAML2.0), Anda dapat mengonfigurasi direktori perusahaan Anda untuk menyediakan akses single-sign on (SSO) ke untuk pengguna Anda. AWS Management Console Untuk informasi selengkapnya, lihat Skenario umum untuk kredensial sementara.

  Jika direktori perusahaan Anda tidak kompatibel dengan SAML 2.0, Anda dapat membuat aplikasi broker identitas untuk menyediakan akses single-sign on (SSO) ke AWS Management Console

untuk pengguna Anda. Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

Jika direktori perusahaan Anda adalah Microsoft Active Directory, Anda dapat menggunakan AWS IAM Identity Center untuk menghubungkan direktori yang dikelola sendiri di Active Directory atau direktori AWS Directory Serviceuntuk membangun kepercayaan antara direktori perusahaan Anda dan direktori Anda Akun AWS.

Jika Anda menggunakan penyedia identitas eksternal (iDP) seperti Okta atau Microsoft Entra untuk mengelola pengguna, Anda dapat menggunakan AWS IAM Identity Center untuk membangun kepercayaan antara IDP Anda dan Anda. Akun AWS Untuk informasi selengkapnya, lihat Connect ke penyedia identitas eksternal di Panduan AWS IAM Identity Center Pengguna.

- Pengguna Anda sudah memiliki identitas Internet.

  Jika Anda membuat aplikasi seluler atau aplikasi berbasis web yang memungkinkan pengguna mengidentifikasi diri mereka melalui penyedia identitas Internet seperti Login with Amazon, Facebook, Google, atau penyedia identitas yang kompatibel dengan OpenID OIDC Connect (), aplikasi dapat menggunakan federasi AWS untuk mengakses. Untuk informasi selengkapnya, lihat OIDCfederasi.

  > **Tip**
  >
  > Untuk menggunakan federasi identitas dengan penyedia identitas Internet, kami sarankan Anda menggunakan Amazon Cognito.

Metode berbeda untuk menyediakan akses pengguna

Berikut adalah cara Anda dapat menyediakan akses ke AWS sumber daya Anda.

Jenis akses pengguna	Kapan itu digunakan?	Dimana informasi lebih lanjut?
Akses masuk tunggal untuk orang-orang, seperti pengguna	IAMIdentity Center menyediakan tempat sentral yang menyatukan administrasi pengguna dan akses mereka	Untuk informasi selengkapnya tentang menyiapkan Pusat IAM Identitas, lihat Memulai di Panduan AWS IAM Identity Center Pengguna

Jenis akses pengguna	Kapan itu digunakan?	Dimana informasi lebih lanjut?
tenaga kerja Anda, ke AWS sumber daya menggunakan Identity Center IAM	ke Akun AWS dan aplikasi cloud. Anda dapat mengatur penyimpanan identitas dalam Pusat IAM Identitas atau Anda dapat mengonfigurasi federasi dengan penyedia identitas yang ada (iDP). Praktik terbaik keamanan merekomendasikan pemberian kredensi terbatas kepada pengguna manusia Anda ke sumber daya. AWS Orang-orang memiliki pengalaman masuk yang lebih mudah dan Anda mempertahankan kendali atas akses mereka ke sumber daya dari satu sistem. IAMIdentity Center mendukung otentikasi multi-faktor (MFA) untuk keamanan akun tambahan.	Untuk informasi selengkapnya tentang penggunaan MFA di Pusat IAM Identitas, lihat Autentikasi multi-faktor di Panduan Pengguna AWS IAM Identity Center

Jenis akses pengguna	Kapan itu digunakan?	Dimana informasi lebih lanjut?
Akses gabungan untuk pengguna manusia, seperti pengguna tenaga kerja Anda, ke AWS layan an yang menggunak an penyedia IAM identitas () IdPs	IAMmendukung IdPs yang kompatibel dengan OpenID Connect (OIDC) atau SAML 2.0 (Security Assertion Markup Language 2.0). Setelah Anda membuat penyedia IAM identitas, buat satu atau beberapa IAM peran yang dapat ditetapkan secara dinamis ke pengguna federasi.	Untuk informasi selengkapnya tentang penyedia IAM identitas dan federasi, lihatPenyedia dan federasi identitas.
Akses lintas akun antara Akun AWS	Anda ingin berbagi akses ke AWS sumber daya tertentu dengan pengguna lain Akun AWS. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, beberapa AWS layanan mendukung kebijakan berbasis sumber daya yang memungkinkan Anda melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy).	Untuk informasi selengkapnya tentang IAM peran, lihatIAMperan. Untuk mengetahui informasi selengkapnya tentang peran terkait layanan, lihat Buat peran tertaut layanan. Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat. AWS layanan yang bekerja dengan IAM Temukan layanan yang memiliki Ya di kolom Peran Tertaut Layanan. Untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut, pilih tautan yang terkait dengan Ya di kolom tersebut.

Jenis akses pengguna	Kapan itu digunakan?	Dimana informasi lebih lanjut?
Kredensi jangka panjang untuk IAM pengguna yang ditunjuk di Akun AWS	Anda mungkin memiliki kasus penggunaan khusus yang memerlukan kredensi jangka panjang dengan IAM pengguna di. AWS Anda dapat menggunakan IAM untuk membuat IAM pengguna ini di Anda Akun AWS, dan gunakan IAM untuk mengelola izin mereka. Beberapa kasus penggunaan meliputi yang berikut: • Beban kerja yang tidak dapat menggunakan peran IAM • AWS Klien pihak ketiga yang memerlukan akses terprogram melalui kunci akses • Kredensi khusus layanan untuk atau Amazon Keyspaces AWS CodeCommit • AWS IAM Identity Center tidak tersedia untuk akun Anda dan Anda tidak memiliki penyedia identitas lain Sebagai praktik terbaik dalam skenario di mana Anda membutuhkan IAM pengguna	Untuk informasi selengkapnya tentang menyiapkan IAM pengguna, lihatBuat IAM pengguna di Akun AWS. Untuk informasi selengkapnya tentang kunci akses IAM pengguna, lihatMengelola kunci akses untuk IAM pengguna. Untuk informasi selengkapnya tentang kredensi khusus layanan untuk atau AWS CodeCommi t Amazon Keyspaces, lihat dan. IAMkredensial untuk: Kredensial CodeCommit Git, SSH kunci, dan kunci akses AWS Gunakan IAM dengan Amazon Keyspaces (untuk Apache Cassandra)

Jenis akses pengguna	Kapan itu digunakan?	Dimana informasi lebih lanjut?
	dengan akses terprogram dan kredensi jangka panjang, kami menyarankan Anda memperbarui kunci akses saat diperlukan. Untuk informasi selengkapnya, lihat Perbarui kunci akses.	

Mendukung akses pengguna terprogram

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS:

- Jika Anda mengelola IAM identitas di Pusat Identitas, AWS APIs memerlukan profil, dan AWS Command Line Interface memerlukan profil atau variabel lingkungan.
- Jika Anda memiliki IAM pengguna, AWS APIs dan AWS Command Line Interface memerlukan kunci akses. Jika memungkinkan, buat kredensial sementara yang terdiri dari ID kunci akses, kunci akses rahasia, dan token keamanan yang menunjukkan masa berlaku kredensial.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses programatis?	Opsi	Informasi lain
Identitas tenaga kerja (Orang dan pengguna dikelola di Pusat IAM Identitas)	Gunakan kredensi jangka pendek untuk menandata ngani permintaan terprogram ke AWS CLI atau AWS APIs (secara langsung atau dengan menggunakan). AWS SDKs	Untuk itu AWS CLI, ikuti petunjuk dalam Mendapatk an kredensi IAM peran untuk CLI akses di AWS IAM Identity Center Panduan Pengguna. Untuk itu AWS APIs, ikuti petunjuk dalam SSOkreden

Pengguna mana yang membutuhkan akses programatis?	Opsi	Informasi lain
		si di Panduan Referensi Alat AWS SDKs dan Alat.
IAMpengguna	Gunakan kredensi jangka pendek untuk menandata ngani permintaan terprogram ke AWS CLI atau AWS APIs (secara langsung atau dengan menggunakan). AWS SDKs	Ikuti petunjuk di Menggunakan kredensial sementara dengan AWS sumber daya.
IAMpengguna	Gunakan kredensi jangka panjang untuk menandata ngani permintaan terprogram ke AWS CLI atau AWS APIs (secara langsung atau dengan menggunakan). AWS SDKs (Tidak direkomendasikan)	Ikuti petunjuk dalam Mengelola kunci akses untuk IAM pengguna.
Pengguna federasi	Gunakan AWS STS API operasi untuk membuat sesi baru dengan kredenal keamanan sementara yang mencakup access key pair dan session token.	Untuk penjelasan API operasi, lihat the section called “Minta kredensi keamanan sementara”

Bagaimana izin dan kebijakan menyediakan manajemen akses

Bagian manajemen akses AWS Identity and Access Management (IAM) membantu Anda menentukan apa yang dapat dilakukan entitas utama dalam akun. Entitas utama adalah orang atau aplikasi yang diautentikasi menggunakan IAM entitas (IAMpengguna atau IAM peran). Manajemen akses sering disebut sebagai otorisasi. Anda mengelola akses AWS dengan membuat kebijakan dan melampirkannya ke IAM identitas (IAMpengguna, IAM grup, atau IAM peran) atau AWS sumber daya.

Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal menggunakan IAM entitas (IAMpengguna atau IAM peran) untuk membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai JSON dokumen. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

Kebijakan dan akun

Jika Anda mengelola satu akun AWS, maka Anda menentukan izin dalam akun tersebut menggunakan kebijakan. Jika Anda mengelola izin di beberapa akun, akan lebih sulit untuk mengelola izin untuk pengguna AndaIAM. Anda dapat menggunakan IAM peran, kebijakan berbasis sumber daya, atau daftar kontrol akses (ACLs) untuk izin lintas akun. Namun, jika Anda memiliki beberapa akun, sebaiknya gunakan AWS Organizations layanan ini untuk membantu Anda mengelola izin tersebut. Untuk informasi lebih lanjut, lihat Apa itu AWS Organizations? dalam Panduan Pengguna Organizations.

Kebijakan dan pengguna

IAMpengguna adalah identitas di. Akun AWS Saat Anda membuat IAM pengguna, mereka tidak dapat mengakses apa pun di akun Anda sampai Anda memberi mereka izin. Anda memberikan izin kepada IAM pengguna dengan membuat kebijakan berbasis identitas, yang merupakan kebijakan yang dilampirkan pada IAM pengguna atau IAM grup tempat pengguna tersebut berada. IAM Contoh berikut menunjukkan JSON kebijakan yang mengizinkan IAM pengguna untuk melakukan semua `dynamodb:*` tindakan Amazon DynamoDB () `Books` pada tabel di akun `123456789012` dalam Wilayah. `us-east-2`

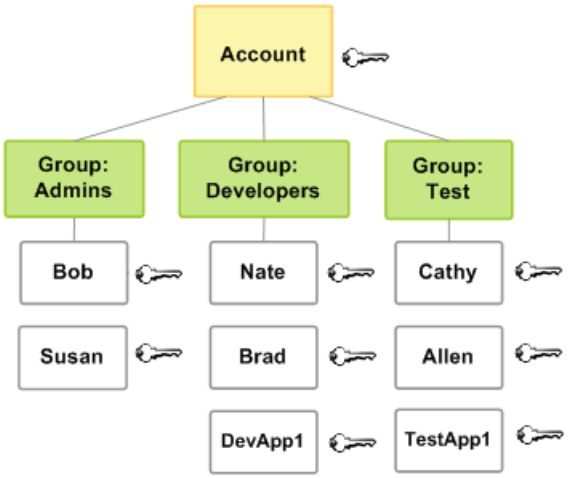
```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Setelah Anda melampirkan kebijakan ini ke IAM pengguna Anda, pengguna memiliki izin untuk melakukan semua tindakan dalam `Books` tabel instance DynamoDB Anda. Sebagian besar IAM pengguna memiliki beberapa kebijakan yang digabungkan untuk mewakili total izin yang diberikan.

Tindakan atau sumber daya yang tidak diizinkan secara eksplisit oleh kebijakan ditolak secara default. Misalnya, jika kebijakan sebelumnya adalah kebijakan tunggal yang dilampirkan ke pengguna, maka pengguna tersebut dapat melakukan tindakan DynamoDB pada `Books` tabel, tetapi tidak dapat melakukan tindakan pada tabel lain. Demikian pula, pengguna tidak diizinkan untuk melakukan tindakan apa pun di AmazonEC2, Amazon S3, atau di AWS layanan lain karena izin untuk bekerja dengan layanan tersebut tidak disertakan dalam kebijakan.

Kebijakan dan IAM grup

Anda dapat mengatur IAM pengguna ke dalam IAMgrup dan melampirkan kebijakan ke IAM grup. Dalam hal ini, IAM pengguna individu masih memiliki kredensialnya sendiri, tetapi semua IAM pengguna dalam IAM grup memiliki izin yang dilampirkan ke grup. IAM Gunakan IAM grup untuk pengelolaan izin yang lebih mudah.



IAMpengguna atau IAM grup dapat memiliki beberapa kebijakan yang dilampirkan padanya yang memberikan izin berbeda. Dalam hal ini, kombinasi kebijakan menentukan izin efektif untuk prinsipal. Jika prinsipal tidak memiliki `Allow` izin eksplisit untuk tindakan dan sumber daya, prinsipal tidak memiliki izin tersebut.

Pengguna gabungan dan peran

Pengguna federasi tidak memiliki identitas permanen seperti Akun AWS yang dilakukan IAM pengguna. Untuk menetapkan izin bagi pengguna gabungan, Anda dapat membuat entitas yang disebut sebagai peran dan menentukan izin untuk peran tersebut. Saat pengguna federasi masuk AWS, pengguna dikaitkan dengan peran tersebut dan diberikan izin yang ditentukan dalam peran tersebut. Untuk informasi selengkapnya, lihat Buat peran untuk penyedia identitas pihak ketiga (federasi).

Kebijakan berbasis identitas dan berbasis sumber daya

Kebijakan berbasis identitas adalah kebijakan izin yang Anda lampirkan ke IAM identitas, seperti IAM pengguna, grup, atau peran. Kebijakan berbasis sumber daya adalah kebijakan izin yang Anda lampirkan ke sumber daya seperti bucket Amazon S3 atau kebijakan kepercayaan peran. IAM

Kebijakan berbasis identitas mengendalikan tindakan apa yang dapat dilakukan oleh identitas, pada sumber daya mana, dan dengan kondisi apa. Kebijakan berbasis identitas selanjutnya dapat dikategorikan menjadi:

- Kebijakan terkelola — Kebijakan berbasis identitas mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Anda. Akun AWS Anda dapat menggunakan dua tipe kebijakan terkelola:
  - AWS kebijakan terkelola — Kebijakan terkelola yang dibuat dan dikelola oleh AWS. Jika Anda baru menggunakan kebijakan, kami sarankan Anda memulai dengan menggunakan kebijakan AWS terkelola.
  - Kebijakan terkelola pelanggan — Kebijakan terkelola yang Anda buat dan kelola di Anda Akun AWS. Kebijakan yang dikelola pelanggan memberikan kontrol yang lebih tepat atas kebijakan Anda daripada kebijakan yang AWS dikelola. Anda dapat membuat, mengedit, dan memvalidasi IAM kebijakan di editor visual atau dengan membuat dokumen JSON kebijakan secara langsung. Untuk informasi selengkapnya, silakan lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan dan Edit IAM kebijakan.
- Kebijakan inline – Kebijakan yang Anda buat dan kelola dan yang disematkan secara langsung ke dalam satu pengguna, grup, atau peran. Dalam kebanyakan kasus, kami tidak menyarankan penggunaan kebijakan inline.

Kebijakan berbasis sumber daya mengontrol tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dengan kondisi seperti apa. Kebijakan berbasis sumber daya merupakan kebijakan inline, dan tidak ada kebijakan berbasis sumber daya terkelola. Untuk mengaktifkan akses lintas akun, Anda dapat menentukan seluruh akun atau IAM entitas di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya.

IAMLayanan ini mendukung satu jenis kebijakan berbasis sumber daya yang disebut kebijakan kepercayaan peran, yang Anda lampirkan ke peran. IAM Karena IAM peran adalah identitas dan sumber daya yang mendukung kebijakan berbasis sumber daya, Anda harus melampirkan kebijakan kepercayaan dan kebijakan berbasis identitas ke peran. IAM Kebijakan kepercayaan menentukan entitas prinsipal mana (akun, pengguna, peran, dan pengguna gabungan) yang dapat memegang

peran tersebut. Untuk mempelajari perbedaan IAM peran dari kebijakan berbasis sumber daya lainnya, lihat. Akses sumber daya lintas akun di IAM

Untuk melihat layanan mana yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari selengkapnya tentang kebijakan berbasis sumber daya, lihat Kebijakan berbasis identitas dan kebijakan berbasis sumber daya.

Tentukan izin berdasarkan atribut dengan otorisasi ABAC

Attribute-based access control (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. AWS memanggil tag atribut ini. Anda dapat melampirkan tag ke IAM sumber daya, termasuk IAM entitas (IAMpengguna atau IAM peran) dan AWS sumber daya. Anda dapat membuat satu ABAC kebijakan atau serangkaian kecil kebijakan untuk IAM prinsipal Anda. Anda dapat mendesain ABAC kebijakan yang memungkinkan operasi saat tag prinsipal cocok dengan tag sumber daya. ABACsistem atribut yang menyediakan konteks pengguna tinggi dan kontrol akses granular. Karena ABAC berbasis atribut, ia dapat melakukan otorisasi dinamis untuk data atau aplikasi yang memberikan atau mencabut akses secara real time. ABACmembantu dalam lingkungan yang skala dan dalam situasi di mana identitas atau manajemen kebijakan sumber daya telah menjadi kompleks.

Misalnya, Anda dapat membuat tiga IAM peran dengan kunci `access-project` tag. Tetapkan nilai tag dari IAM peran pertama ke`Heart`, yang kedua ke`Star`, dan yang ketiga ke`Lightning`. Anda kemudian dapat menggunakan kebijakan tunggal yang memungkinkan akses ketika IAM peran dan AWS sumber daya memiliki nilai tag`access-project`. Untuk tutorial terperinci yang menunjukkan ABAC cara menggunakannya AWS, lihatIAMtutorial: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag. Untuk mempelajari tentang layanan yang mendukungABAC, lihatAWS layanan yang bekerja dengan IAM.

Perbandingan ABAC dengan RBAC model tradisional

Model otorisasi tradisional yang digunakan IAM adalah kontrol akses berbasis peran (). RBAC RBACmendefinisikan izin berdasarkan fungsi pekerjaan seseorang, atau peran, yang berbeda dari peranIAM. IAMmemang menyertakan kebijakan terkelola untuk fungsi pekerjaan yang menyelaraskan izin ke fungsi pekerjaan dalam suatu RBAC model.

DiIAM, Anda menerapkan RBAC dengan membuat kebijakan yang berbeda untuk fungsi pekerjaan yang berbeda. Anda kemudian melampirkan kebijakan ke identitas (IAMpengguna, IAM grup, atau IAM peran). Sebagai praktik terbaik, Anda memberikan izin minimum yang diperlukan untuk fungsi pekerjaan. Ini menghasilkan akses hak istimewa paling sedikit. Setiap kebijakan fungsi pekerjaan mencantumkan sumber daya spesifik yang dapat diakses oleh identitas yang ditetapkan kebijakan

tersebut. Kerugian menggunakan RBAC model tradisional adalah ketika Anda atau pengguna menambahkan sumber daya baru ke lingkungan Anda, Anda harus memperbarui kebijakan untuk mengizinkan akses ke sumber daya tersebut.

Misalnya, anggaplah Anda memiliki tiga proyek bernama `Heart`, `Star`, dan `Lightning`, yang dikerjakan karyawan Anda. Anda membuat IAM peran untuk setiap proyek. Anda kemudian melampirkan kebijakan ke setiap IAM peran untuk menentukan sumber daya yang dapat diakses oleh siapa pun yang diizinkan untuk mengambil IAM peran. Jika seorang karyawan berganti pekerjaan di perusahaan Anda, Anda menugaskan mereka ke IAM peran yang berbeda. Anda dapat menetapkan orang atau program untuk lebih dari satu IAM peran. Namun, `Star` proyek ini mungkin memerlukan sumber daya tambahan, seperti EC2 wadah Amazon baru. Dalam hal ini, Anda harus memperbarui kebijakan yang dilampirkan ke `Star` IAM peran untuk menentukan sumber daya penampung baru. Jika tidak, anggota `Star` proyek tidak diizinkan mengakses wadah baru.

ABACmemberikan keuntungan sebagai berikut dibandingkan RBAC model tradisional:

- ABACskala izin dengan inovasi. Administrator idak perlu lagi memperbarui kebijakan yang ada untuk memungkinkan akses ke sumber daya baru. Misalnya, asumsikan bahwa Anda merancang ABAC strategi Anda dengan `access-project` tag. Pengembang menggunakan IAM peran dengan `Heart` tag `access-project` =. Ketika orang-orang di `Heart` proyek membutuhkan EC2 sumber daya Amazon tambahan, pengembang dapat membuat EC2 instance Amazon baru dengan `Heart` tag `access-project` =. Lalu, siapa pun di proyek `Heart` dapat memulai dan menghentikan instans tersebut karena nilai tanda mereka cocok.
- ABACMembutuhkan kebijakan yang lebih sedikit. Karena Anda tidak perlu membuat kebijakan yang berbeda untuk fungsi pekerjaan yang berbeda, Anda membuat kebijakan yang lebih sedikit. Kebijakan tersebut lebih mudah dikelola.
- Dengan menggunakanABAC, tim dapat merespons perubahan dan pertumbuhan secara dinamis. Karena izin untuk sumber daya baru secara otomatis diberikan berdasarkan atribut, Anda tidak perlu menetapkan kebijakan identitas secara manual. Misalnya, jika perusahaan Anda sudah mendukung `Heart` dan `Star` proyek menggunakanABAC, mudah untuk menambahkan `Lightning` proyek baru. IAMAdministrator membuat IAM peran baru dengan `Lightning` tag `access-project` =. Tidak perlu mengubah kebijakan untuk mendukung proyek baru. Siapa pun yang memiliki izin untuk mengambil IAM peran dapat membuat dan melihat instance yang ditandai dengan =. `access-project` `Lightning` Skenario lain adalah ketika seorang anggota tim pindah dari `Heart` proyek ke `Lightning` proyek. Untuk memberikan akses anggota tim ke `Lightning` proyek, IAM administrator menugaskan mereka ke IAM peran yang berbeda. Tidak perlu mengubah kebijakan izin.

- Izin granular dimungkinkan menggunakan. ABAC Saat Anda membuat kebijakan, memberikan hak istimewa terkecil adalah praktik terbaik. Menggunakan tradisionalRBAC, Anda menulis kebijakan yang memungkinkan akses ke sumber daya tertentu. Namun, saat Anda menggunakanABAC, Anda dapat mengizinkan tindakan pada semua sumber daya jika tag sumber daya cocok dengan tag prinsipal.
- Gunakan atribut karyawan dari direktori perusahaan Anda denganABAC. Anda dapat mengonfigurasi SAML atau OIDC penyedia Anda untuk meneruskan tag sesiIAM. Ketika karyawan Anda bergabung AWS, IAM terapkan atribut mereka ke kepala sekolah yang dihasilkan. Anda kemudian dapat menggunakan ABAC untuk mengizinkan atau menolak izin berdasarkan atribut tersebut.

Untuk tutorial terperinci yang menunjukkan ABAC cara menggunakannya AWS, lihatIAMtutorial: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag.

Memulai dengan IAM

AWS Identity and Access Management (IAM) membantu Anda mengontrol akses ke Amazon Web Services (AWS) dan sumber daya akun Anda dengan aman. IAMjuga dapat menjaga kerahasiaan login Anda. Anda tidak secara khusus mendaftar untuk digunakanIAM. Tidak ada biaya untuk digunakanIAM.

Gunakan IAM untuk memberikan identitas, seperti pengguna dan peran, akses ke sumber daya di akun Anda. Misalnya, Anda dapat menggunakan IAM dengan pengguna yang ada di direktori perusahaan yang Anda kelola eksternal AWS atau Anda dapat membuat pengguna dalam AWS menggunakan AWS IAM Identity Center. Identitas federasi mengambil IAM peran yang ditentukan untuk mengakses sumber daya yang mereka butuhkan. Untuk informasi selengkapnya tentang Pusat IAM Identitas, lihat Apa itu Pusat IAM Identitas? dalam AWS IAM Identity Center User Guide.

> **Note**
>
> IAMterintegrasi dengan beberapa AWS produk. Untuk daftar layanan yang mendukungIAM, lihatAWS layanan yang bekerja dengan IAM.

Untuk mempelajari cara memulai AWS, membuat pengguna administratif, Organizations, dan menggunakan beberapa layanan untuk memecahkan masalah seperti membangun dan meluncurkan proyek pertama Anda, lihat Pusat Sumber Daya Memulai.

Menyiapkan Anda Akun AWS

Sebelum Anda mulai bekerja denganIAM, pastikan Anda telah menyelesaikan pengaturan awal AWS lingkungan Anda.

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan tindakan menerima panggilan telepon dan memasukkan kode verifikasi di keypad telepon.

> Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimkan email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Ketika Anda mendaftar untuk layanan, Anda membuat Akun AWS menggunakan alamat email dan kata sandi. Itu adalah kredensi pengguna AWS root Anda. Sebagai praktik terbaik, Anda tidak menggunakan kredensi pengguna root Anda AWS untuk mengakses tugas sehari-hari. Hanya gunakan kredensi pengguna root Anda untuk melakukan tugas yang memerlukan kredensi pengguna root. Juga, jangan berbagi kredensi Anda dengan orang lain. Sebagai gantinya, tambahkan orang ke direktori Anda dan beri mereka akses ke direktori Anda Akun AWS.

Untuk mengamankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.

2. Aktifkan otentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan MFA perangkat virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan IAM Pengguna.

Berikan akses ke konsol penagihan

IAMpengguna dan peran dalam tidak Akun AWS dapat mengakses konsol Billing and Cost Management secara default. Ini benar bahkan jika mereka memiliki IAM kebijakan yang memberikan akses ke fitur Penagihan tertentu. Untuk memberikan akses, pengguna Akun AWS root harus terlebih dahulu mengaktifkan IAM akses.

> **Note**
>
> Sebagai praktik terbaik keamanan, kami menyarankan Anda menyediakan akses ke sumber daya Anda melalui federasi identitas dengan AWS IAM Identity Center. Saat Anda mengaktifkan Pusat IAM Identitas bersama dengan AWS Organizations, konsol Billing and Cost Management diaktifkan secara default dengan penagihan gabungan untuk Akun AWS semua di organisasi Anda. Untuk informasi selengkapnya, lihat Mengkonsolidasikan penagihan AWS Organizations di Panduan Pengguna Billing and Cost Management.

1. Masuk ke AWS Management Console dengan kredensi pengguna root Anda (khususnya, alamat email dan kata sandi yang Anda gunakan untuk membuat AWS akun Anda).
2. Pada bilah navigasi, pilih nama akun Anda, lalu pilih Akun.
3. Gulir ke bawah halaman hingga Anda menemukan bagian Akses IAM Pengguna dan Peran ke Informasi Penagihan, lalu pilih Edit.
4. Pilih kotak centang Aktifkan IAM Akses untuk mengaktifkan akses ke halaman konsol Billing and Cost Management.
5. Pilih Perbarui.

   Halaman menampilkan pesan IAMpengguna/akses peran ke informasi penagihan diaktifkan.

   > **Important**
   >
   > Mengaktifkan IAM akses saja tidak memberikan izin apa pun bagi pengguna atau peran untuk melihat halaman konsol Billing and Cost Management. Anda juga harus melampirkan kebijakan berbasis identitas yang diperlukan ke IAM peran untuk memberikan akses ke konsol penagihan. Peran menyediakan kredensi sementara yang dapat diasumsikan pengguna saat diperlukan.

6. Gunakan AWS Management Console untuk membuat peran yang dapat diasumsikan pengguna untuk mengakses konsol penagihan.
7. Pada halaman Tambahkan izin untuk peran tersebut, tambahkan izin ke daftar dan lihat detail tentang sumber daya Penagihan di situs Anda. Akun AWS

   Kebijakan AWS terkelola Penagihan memberi pengguna izin untuk melihat dan mengedit konsol Billing and Cost Management. Ini termasuk melihat penggunaan akun, memodifikasi anggaran dan metode pembayaran. Untuk contoh kebijakan lainnya yang dapat Anda lampirkan ke IAM

peran untuk mengontrol akses ke informasi penagihan akun Anda, lihat contoh kebijakan AWS penagihan di Panduan Pengguna Billing and Cost Management.

Untuk membuat pengguna dengan akses administratif

1. Aktifkan Pusat IAM Identitas.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat IAM Identitas, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat IAM Identitas, gunakan login URL yang dikirim ke alamat email saat Anda membuat pengguna Pusat IAM Identitas.

  Untuk bantuan masuk menggunakan pengguna Pusat IAM Identitas, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat IAM Identitas, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuk, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Pengaturan IAM awal untuk akun Anda

AWS Identity and Access Management adalah AWS layanan dasar, yang membantu Anda mengelola akses ke sumber daya Anda AWS dengan aman. Mengelola IAM menyediakan berbagai tanggung

jawab untuk mengontrol akses dan izin pengguna, mulai dari mendukung berbagai jenis pengguna hingga mengelola kata sandi, izin, dan kredensi keamanan.

Anda dapat menggunakan IAM layanan tambahan di lingkungan Anda untuk membantu Anda dengan identitas dan tujuan manajemen akses Anda. Aktifkan AWS IAM Identity Center dengan Organizations untuk membuat instance organisasi dari IAM Identity Center untuk mengelola akses orang dan grup secara terpusat ke aplikasi dan Akun AWS. Gunakan IAM Access Analyzer untuk membantu Anda mengelola izin di akun dan organisasi Anda dengan meninjau temuan akses. Untuk mendapatkan kredensi keamanan sementara IAM untuk beban kerja seperti server, kontainer, dan aplikasi yang berjalan di luar AWS, gunakan IAM Peran Di Mana Saja.

Ketika Anda awalnya mengatur AWS lingkungan Anda, Anda membuat keputusan tentang:

- Yang URL Anda gunakan untuk terhubung AWS. Ini URL didasarkan pada Akun AWS id Anda. Untuk informasi selengkapnya, lihat Melihat Akun AWS ID Anda. Agar URL lebih mudah diingat, Anda dapat mengonfigurasi alias untuk akun Anda.
- Bagaimana Anda akan mengatur identitas di lingkungan Anda. Anda dapat menggunakan Pusat IAM Identitas untuk menambahkan orang ke direktori organisasi, mengkategorikannya ke dalam grup, dan memberi mereka akses ke aplikasi dan sumber daya. Anda dapat menggabungkan Pusat IAM Identitas atau IAM dengan penyedia identitas eksternal untuk mengintegrasikan sumber identitas Anda yang ada. AWS
- Izin apa yang diperlukan untuk melakukan tugas yang berbeda. Izin dikendalikan melalui kebijakan yang dapat diterapkan langsung ke IAM peran atau yang dapat diterapkan secara otomatis ke IAM peran yang dibuat mengapa Anda menggunakan Pusat IAM Identitas untuk membuat kumpulan izin.
- IAMPeran mana yang perlu Anda dukung di lingkungan Anda. Ada beberapa skenario umum yang melibatkan IAM peran yang memenuhi persyaratan akses yang berbeda.

Saat Anda membuat IAM komponen yang berbeda dari identitas dan sistem manajemen akses Anda, Anda mungkin perlu merujuk kembali ke item lain yang telah Anda kerjakan di lingkungan Anda. IAMmenyediakan fitur pencarian untuk membantu Anda menemukan sesuatu dengan cepat dan mudah.

Melihat Akun AWS ID Anda

Jika Anda masuk ke konsol, Anda dapat melihat ID akun untuk Anda Akun AWS menggunakan metode berikut.

Untuk melihat Akun AWS ID Anda

IAM console

ID AWS akun ditampilkan saat Anda pergi ke IAM Dasbor di Akun AWS bagian. Ada cara tambahan untuk melihat ID akun Anda di konsol tergantung pada jenis pengguna Anda. Jika Anda telah mengambil peran, kredensi keamanan tidak tersedia.

Jenis pengguna	Prosedur
Pengguna root	Di bilah navigasi di kanan atas, pilih nama pengguna Anda dan kemudian pilih Kredensi keamanan. Nomor akun muncul di bawah Pengidentifikasi akun.
IAMpengguna	Di bilah navigasi di kanan atas, pilih nama pengguna Anda, ID akun ditampilkan di atas nama pengguna Anda. Pilih Kredensia l keamanan. Nomor akun muncul di bawah Detail akun.
Pengguna federasi	Di bilah navigasi di kanan atas, pilih nama pengguna Anda, ID akun ditampilkan di atas nama pengguna Anda.
Asumsi peran	Di bilah navigasi di kanan atas, pilih ikon Support, lalu pilih Support Center dari daftar. Nomor akun (ID) 12 digit Anda yang masuk saat ini muncul di panel navigasi Pusat Dukungan.

AWS CLI

Gunakan perintah berikut untuk melihat ID pengguna, ID akun, dan pengguna AndaARN:

- aws sts get-caller-identity

API

Gunakan berikut ini API untuk melihat ID pengguna, ID akun, dan pengguna AndaARN:

- GetCallerIdentity

Menggunakan alias untuk ID Anda Akun AWS

ID akun Anda adalah nomor 12 digit yang secara unik mengidentifikasi akun Anda. Secara default, IAM pengguna di akun masuk menggunakan web URL yang menyertakan ID akun. Jika mereka tidak memilikiURL, mereka dapat memberikan ID akun di halaman AWS login saat mereka masuk.

Halaman login Anda URL memiliki format berikut, secara default.

```
https://Your_Account_ID.signin.aws.amazon.com/console/
```

Banyak orang menganggap kata-kata lebih mudah diingat daripada angka, jadi membuat alias untuk ID akun Anda dapat membantu IAM pengguna Anda masuk lebih mudah.

Jika Anda membuat Akun AWS alias untuk Akun AWS ID Anda, halaman login Anda URL terlihat seperti contoh berikut.

```
https://Your_Account_Alias.signin.aws.amazon.com/console/
```

Pertimbangan sebelum membuat alias akun

- Anda hanya Akun AWS dapat memiliki satu alias. Jika Anda membuat alias baru untuk AWS akun Anda, alias baru menimpa alias sebelumnya, dan yang URL berisi alias sebelumnya berhenti berfungsi.
- Alias akun harus berisi hanya digit, huruf kecil, dan tanda hubung. Untuk informasi selengkapnya tentang batasan entitas AWS akun, lihatIAMdan AWS STS kuota.
- Alias akun harus unik di semua produk Amazon Web Services dalam partisi jaringan tertentu.

  Partisi adalah grup Wilayah AWS . Setiap AWS akun dicakup ke satu partisi.

  Berikut ini adalah partisi yang didukung:

  - `aws`- AWS Wilayah
  - `aws-cn` - Wilayah Tiongkok
  - `aws-us-gov` - AWS GovCloud (US) Wilayah

> **Note**
>
> Alias akun bukanlah rahasia, dan mereka akan muncul di halaman login Anda yang menghadap publik. URL Jangan sertakan informasi sensitif apa pun dalam alias akun Anda. Asli URL yang berisi Akun AWS ID Anda tetap aktif dan dapat digunakan setelah Anda membuat Akun AWS alias Anda.

Membuat alias akun

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:

- `iam:ListAccountAliases`
- `iam:CreateAccountAlias`

Untuk membuat Akun AWS alias

Pilih tab untuk metode yang ingin Anda ikuti untuk membuat alias akun:

IAM console

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Dasbor.
3. Di bagian AWS Akun, di samping Alias Akun, pilih Buat. Jika alias sudah ada, maka pilih Edit.
4. Di kotak dialog, masukkan nama yang ingin Anda gunakan untuk alias Anda, lalu pilih Simpan perubahan.

AWS CLI

Jalankan perintah berikut:

- `aws iam create-account-alias`

API

Untuk membuat alias untuk halaman AWS Management Console login AndaURL, panggil operasi berikut:

- `CreateAccountAlias`

Menghapus alias akun

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:

- `iam:ListAccountAliases`
- `iam:DeleteAccountAlias`

Untuk menghapus alias akun

Pilih tab untuk metode yang ingin Anda ikuti untuk menghapus alias akun:

IAM console

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Dasbor.
3. Di bagian AWS Akun, di samping Alias Akun, pilih Hapus.

AWS CLI

Untuk menghapus alias Akun AWS ID, jalankan perintah berikut:

- `aws iam delete-account-alias`

Untuk mengonfirmasi bahwa alias akun dihapus, coba tampilkan alias Akun AWS ID Anda, dengan menjalankan perintah berikut:

- `aws iam list-account-aliases`

API

Untuk menghapus alias Akun AWS ID, panggil operasi berikut:

- `DeleteAccountAlias`

Untuk mengonfirmasi bahwa alias akun dihapus, cobalah untuk menampilkan alias Akun AWS ID Anda, dengan memanggil operasi berikut:

- `ListAccountAliases`

> **Note**
>
> Setelah menghapus alias akun Anda, satu-satunya login URL untuk akun Anda didasarkan pada ID akun Anda. Setiap upaya untuk terhubung ke alias URL akan gagal dan tidak dialihkan.

Rencanakan akses ke AWS akun Anda

Saat menyiapkan AWS, rencanakan bagaimana Anda ingin orang mengakses AWS akun dan sumber daya Anda untuk menyiapkan solusi manajemen identitas yang dirancang dengan baik dan aman.

Sumber identitas

Menurut praktik IAM terbaik, pengguna manusia dan beban kerja harus menggunakan kredensi sementara ketika mereka mengakses sumber daya Anda. AWS Kredensi sementara diberikan kepada identitas yang mengakses sumber daya Anda menggunakan peran. IAM Kedua pengguna federasi ke dalam IAM dan pengguna di Pusat IAM Identitas (baik federasi atau dibuat di direktori Pusat IAM Identitas) menggunakan IAM peran untuk mengakses sumber daya.

Sebelum Anda mulai menggunakan AWS, rencanakan cara mengatur identitas Anda baik dengan:

- Mengaktifkan Pusat IAM Identitas dengan Organizations dan menambahkan pengguna di Pusat IAM Identitas langsung ke direktori organisasi.

  Untuk mempelajari cara menambahkan pengguna secara langsung ke direktori organisasi Pusat IAM Identitas, lihat Menambahkan pengguna
- Menggabungkan penyedia identitas eksternal Anda yang ada dengan Pusat IAM Identitas ataulAM.

  Untuk mempelajari cara menggabungkan penyedia identitas eksternal ke direktori organisasi Pusat IAM Identitas, gunakan tutorial Memulai yang sesuai.

Manajemen akses

Identifikasi AWS sumber daya dan layanan yang akan diakses pengguna Anda dan tentukan izin akses dan kebijakan yang diperlukan untuk setiap pengguna, grup, atau peran.

- Jika Anda menggunakan Pusat IAM Identitas, penyedia IAM identitas serta kebijakan IAM peran dan izin secara otomatis dibuat di setiap AWS akun di organisasi Anda. Peran dan izin ini selaras dengan izin yang Anda tentukan saat Anda menetapkan orang atau grup ke aplikasi atau akun tertentu. AWS

  Untuk informasi selengkapnya, lihat Menetapkan akses pengguna dan Mengatur akses masuk tunggal ke aplikasi Anda.
- Jika Anda mengfederasikan penyedia identitas Anda secara langsung Akun AWS, Anda harus membuat peran bagi pengguna untuk diasumsikan dan dua kebijakan; kebijakan kepercayaan yang menentukan siapa yang dapat mengambil peran tersebut, dan kebijakan izin yang menentukan AWS tindakan dan sumber daya yang diizinkan atau ditolak aksesnya oleh orang yang mengambil peran tersebut. IAM

  Untuk informasi selengkapnya, lihat Penyedia dan federasi identitas

Kasus penggunaan untuk IAM pengguna

IAMpengguna yang Anda buat di Anda Akun AWS memiliki kredensi jangka panjang yang Anda kelola secara langsung.

Ketika datang untuk mengelola akses di AWS, IAM pengguna umumnya bukan pilihan terbaik. Ada beberapa alasan utama mengapa Anda harus menghindari mengandalkan IAM pengguna untuk sebagian besar kasus penggunaan Anda.

Pertama, IAM pengguna dirancang untuk akun individual, sehingga mereka tidak menskalakan dengan baik seiring pertumbuhan organisasi Anda. Mengelola izin dan keamanan untuk sejumlah besar IAM pengguna dapat dengan cepat menjadi tantangan.

IAMpengguna juga tidak memiliki visibilitas terpusat dan kemampuan audit yang Anda dapatkan dengan solusi manajemen AWS identitas lainnya. Ini dapat membuatnya lebih menantang untuk menjaga keamanan dan kepatuhan terhadap peraturan.

Akhirnya, menerapkan praktik terbaik keamanan seperti otentikasi multi-faktor, kebijakan kata sandi, dan pemisahan peran jauh lebih mudah dengan pendekatan manajemen identitas yang lebih skalabel.

Alih-alih mengandalkan IAM pengguna, sebaiknya gunakan solusi yang lebih kuat seperti Pusat IAM Identitas dengan AWS Organizations, atau identitas gabungan dari penyedia eksternal. Opsi ini akan memberi Anda kontrol, keamanan, dan efisiensi operasional yang lebih baik seiring pertumbuhan AWS lingkungan Anda.

Oleh karena itu, kami menyarankan Anda hanya menggunakan IAM pengguna untuk kasus penggunaan yang tidak didukung oleh pengguna federasi.

Daftar berikut mengidentifikasi kasus penggunaan spesifik yang memerlukan kredensi jangka panjang dengan IAM pengguna di. AWS Anda dapat menggunakan IAM untuk membuat IAM pengguna ini di bawah payung AWS akun Anda, dan gunakan IAM untuk mengelola izin mereka.

- Akses darurat ke AWS akun Anda
- Beban kerja yang tidak dapat menggunakan peran IAM
  - AWS CodeCommit akses
  - Akses Amazon Keyspaces (untuk Apache Cassandra)
- AWS Klien pihak ketiga
- AWS IAM Identity Center tidak tersedia untuk akun Anda dan Anda tidak memiliki penyedia identitas lain

Buat IAM pengguna untuk akses darurat

IAMPengguna adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi.

Memiliki IAM pengguna untuk akses darurat adalah salah satu alasan yang disarankan untuk membuat IAM pengguna sehingga Anda dapat mengakses Akun AWS jika penyedia identitas Anda tidak dapat diakses.

> **Note**
>
> Sebagai praktik keamanan terbaik, kami menyarankan Anda menyediakan akses ke sumber daya Anda melalui federasi identitas alih-alih membuat IAM pengguna. Untuk informasi

tentang situasi tertentu di mana IAM pengguna diperlukan, lihat Kapan membuat IAM pengguna (bukan peran).

Untuk membuat IAM pengguna untuk akses darurat

Pilih tab untuk metode yang ingin Anda ikuti untuk membuat IAM pengguna:

Izin minimum

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:

- `access-analyzer:ValidatePolicy`
- `iam:AddUserToGroup`
- `iam:AttachGroupPolicy`
- `iam:CreateGroup`
- `iam:CreateLoginProfile`
- `iam:CreateUser`
- `iam:GetAccountPasswordPolicy`
- `iam:GetLoginProfile`
- `iam:GetUser`
- `iam:ListAttachedGroupPolicies`
- `iam:ListAttachedUserPolicies`
- `iam:ListGroupPolicies`
- `iam:ListGroups`
- `iam:ListGroupsForUser`
- `iam:ListPolicies`
- `iam:ListUserPolicies`
- `iam:ListUsers`

IAM console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.

2. Di halaman Beranda Konsol, pilih IAM layanan.
3. Di panel navigasi, pilih Pengguna dan kemudian pilih Buat pengguna.

> **Note**
>
> Jika Anda mengaktifkan Pusat IAM Identitas, akan AWS Management Console menampilkan pengingat bahwa yang terbaik adalah mengelola akses pengguna di Pusat IAM Identitas. Dalam prosedur ini, IAM pengguna yang Anda buat khusus digunakan hanya jika penyedia identitas Anda tidak tersedia.

4. Pada halaman Tentukan detail pengguna, di bawah Rincian pengguna, di Nama pengguna, masukkan nama untuk pengguna baru. Ini adalah nama masuk mereka untuk AWS. Untuk contoh ini, masukkan**`EmergencyAccess`**.

> **Note**
>
> Nama pengguna dapat berupa kombinasi hingga 64 huruf, digit, dan karakter ini: plus (+), sama dengan (=), koma (,), titik (.), pada a keong (@), garis bawah (_), dan tanda hubung (-). Nama harus unik dalam akun. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat dua pengguna bernama TESTUSER dan testuser. Ketika nama pengguna digunakan dalam kebijakan atau sebagai bagian dariARN, nama tersebut peka huruf besar/kecil. Ketika nama pengguna muncul ke pelanggan di konsol, seperti selama proses login, nama pengguna tidak peka huruf besar/kecil.

5. Pilih kotak centang di sebelah Berikan akses pengguna ke AWS Management Console—opsional dan kemudian pilih Saya ingin membuat IAM pengguna.
6. Di bawah Kata sandi konsol, pilih Kata sandi yang dibuat otomatis.
7. Kosongkan kotak centang di sebelah Pengguna harus membuat kata sandi baru saat masuk berikutnya (disarankan). Karena IAM pengguna ini untuk akses darurat, administrator tepercaya mempertahankan kata sandi dan hanya menyediakannya bila diperlukan.
8. Pada halaman Setel izin, di bawah opsi Izin, pilih Tambahkan pengguna ke grup. Kemudian, di bawah Grup pengguna, pilih Buat grup.
9. Pada halaman Buat grup pengguna, di Nama grup pengguna, masukkan**`EmergencyAccessGroup`**. Kemudian, di bawah Kebijakan izin, pilih AdministratorAccess.

10. Pilih Buat grup pengguna untuk kembali ke halaman Setel izin.
11. Di bawah Grup pengguna, pilih nama yang **`EmergencyAccessGroup`** Anda buat sebelumnya.
12. Pilih Berikutnya untuk melanjutkan ke halaman Tinjauan dan buat.
13. Pada halaman Tinjau dan buat, tinjau daftar keanggotaan grup pengguna yang akan ditambahkan ke pengguna baru. Ketika Anda siap untuk melanjutkan, pilih Buat pengguna.
14. Pada halaman Ambil kata sandi, pilih Unduh file.csv untuk menyimpan file.csv dengan informasi kredensi pengguna (KoneksiURL, nama pengguna, dan kata sandi).
15. Simpan file ini untuk digunakan jika Anda perlu masuk IAM dan tidak memiliki akses ke penyedia identitas Anda.

IAMPengguna baru ditampilkan dalam daftar Pengguna. Pilih tautan Nama pengguna untuk melihat detail pengguna.

AWS CLI

1. Buat pengguna bernama**`EmergencyAccess`**.

   - aws iam buat-pengguna

   ```
   aws iam create-user \
      --user-name EmergencyAccess
   ```

2. (Opsional) Berikan akses pengguna ke AWS Management Console. Ini memerlukan kata sandi. Untuk membuat kata sandi bagi IAM pengguna, Anda dapat menggunakan `--cli-input-json` parameter untuk meneruskan JSON file yang berisi kata sandi.Anda juga harus memberi pengguna halaman URL masuk akun Anda.

   - aws iam create-login-profile

   ```
   aws iam create-login-profile \
      --generate-cli-skeleton > create-login-profile.json
   ```

   - Buka `create-login-profile.json` file di editor teks dan masukkan kata sandi yang sesuai dengan kebijakan kata sandi Anda, lalu simpan file tersebut. Sebagai contoh:

```
{
 "UserName": "EmergencyAccess",
 "Password": "Ex@3dRA0djs",
 "PasswordResetRequired": false
}
```

- Gunakan `aws iam create-login-profile` perintah lagi, lewati `--cli-input-json` parameter untuk menentukan JSON file Anda.

```
aws iam create-login-profile \
   --cli-input-json file://create-login-profile.json
```

> **Note**
>
> Jika kata sandi yang Anda berikan dalam JSON file melanggar kebijakan kata sandi akun Anda, Anda akan menerima `PassworPolicyViolation` kesalahan. Jika ini terjadi, tinjau kebijakan kata sandi untuk akun Anda dan perbarui kata sandi dalam JSON file untuk memenuhi persyaratan.

3. Buat**`EmergencyAccessGroup`**, lampirkan kebijakan AWS terkelola `AdministratorAccess` ke grup, dan tambahkan **`EmergencyAccess`** pengguna ke grup.

> **Note**
>
> Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Setiap kebijakan memiliki Nama Sumber Daya Amazon (ARN) sendiri yang menyertakan nama kebijakan. Misalnya, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` adalah kebijakan yang AWS dikelola. Untuk informasi lebih lanjut tentangARNs, lihatIAM ARNs. Untuk mengetahui daftar kebijakan AWS terkelola Layanan AWS, lihat kebijakan AWS terkelola.

- aws iam membuat grup

```
aws iam create-group \
   --group-name EmergencyAccessGroup
```

- aws iam attach-group-policy

```
aws iam attach-group-policy \
   --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
   --group-name >EmergencyAccessGroup
```

- aws iam add-user-to -grup

```
aws iam add-user-to-group \
   --user-name EmergencyAccess \
   --group-name EmergencyAccessGroup
```

- Jalankan perintah aws iam get-group untuk mencantumkan **EmergencyAccessGroup** dan anggotanya.

```
aws iam get-group \
   --group-name EmergencyAccessGroup
```

Buat IAM pengguna untuk beban kerja yang tidak dapat menggunakan peran IAM

Important

Sebagai praktik terbaik, kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Anda dapat menggunakan penyedia identitas bagi pengguna manusia Anda untuk menyediakan akses federasi Akun AWS dengan mengambil peran, yang menyediakan kredensi sementara. Untuk manajemen akses terpusat, kami menyarankan Anda menggunakan AWS IAM Identity Center (Pusat IAM Identitas) untuk mengelola akses ke akun Anda dan izin dalam akun tersebut. Anda dapat membuat dan mengelola identitas pengguna Anda, termasuk pengguna administratif Anda, dengan Pusat IAM Identitas. Jika Anda menggunakan penyedia identitas eksternal, Anda

juga dapat mengonfigurasi izin akses untuk identitas pengguna di IAM Pusat Identitas. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center dalam Panduan Pengguna AWS IAM Identity Center .

Jika kasus penggunaan Anda mengharuskan IAM pengguna dengan akses terprogram dan kredensi jangka panjang, sebaiknya Anda membuat prosedur untuk memperbarui kunci akses bila diperlukan. Untuk informasi selengkapnya, lihat Perbarui kunci akses.

Untuk melakukan beberapa tugas manajemen akun dan layanan, Anda harus masuk menggunakan kredensi pengguna root. Untuk melihat tugas yang mengharuskan Anda masuk sebagai pengguna root, lihatTugas yang memerlukan kredensial pengguna root.

Untuk membuat IAM pengguna untuk beban kerja yang tidak dapat menggunakan peran IAM

Pilih tab untuk metode yang ingin Anda ikuti untuk membuat IAM pengguna untuk beban kerja:

Izin minimum

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:

- `iam:AddUserToGroup`
- `iam:AttachGroupPolicy`
- `iam:CreateAccessKey`
- `iam:CreateGroup`
- `iam:CreateServiceSpecificCredential`
- `iam:CreateUser`
- `iam:GetAccessKeyLastUsed`
- `iam:GetAccountPasswordPolicy`
- `iam:GetAccountSummary`
- `iam:GetGroup`
- `iam:GetLoginProfile`
- `iam:GetPolicy`
- `iam:GetRole`
- `iam:GetUser`
- `iam:ListAccessKeys`

- `iam:ListAttachedGroupPolicies`
- `iam:ListAttachedUserPolicies`
- `iam:ListGroupPolicies`
- `iam:ListGroups`
- `iam:ListGroupsForUser`
- `iam:ListInstanceProfilesForRole`
- `iam:ListMFADevices`
- `iam:ListPolicies`
- `iam:ListRoles`
- `iam:ListRoleTags`
- `iam:ListSSHPublicKeys`
- `iam:ListServiceSpecificCredentials`
- `iam:ListSigningCertificates`
- `iam:ListUserPolicies`
- `iam:ListUserTags`
- `iam:ListUsers`
- `iam:UploadSSHPublicKey`
- `iam:UploadSigningCertificate`

IAM console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Di halaman Beranda Konsol, pilih IAM layanan.
3. Di panel navigasi, pilih Pengguna dan kemudian pilih Buat pengguna.
4. Pada halaman Tentukan detail pengguna, lakukan hal berikut:
   a. Untuk Nama pengguna, ketik ***`WorkloadName`***. Ganti ***`WorkloadName`***dengan nama beban kerja yang akan menggunakan akun.
   b. Pilih Berikutnya.
5. (Opsional) Pada halaman Set Permissions, lakukan hal berikut:

a. Pilih Tambahkan pengguna ke grup.

b. Pilih Buat kelompok.

c. Dalam kotak dialog Buat grup pengguna, untuk Nama grup pengguna ketikkan nama yang mewakili penggunaan beban kerja dalam grup. Untuk contoh ini, gunakan nama**`Automation`**.

d. Di bawah Kebijakan izin, pilih kotak centang untuk kebijakan PowerUserAccessterkelola.

> **Tip**
>
> Masukkan Daya ke dalam kotak pencarian Kebijakan izin untuk menemukan kebijakan terkelola dengan cepat.

e. Pilih Buat grup pengguna.

f. Kembali ke halaman dengan daftar IAM grup, pilih kotak centang untuk grup pengguna baru Anda. Pilih Refresh (Segarkan) jika Anda tidak melihat grup pengguna baru di dalam daftar.

g. Pilih Berikutnya.

6. (Opsional) Di bagian Tag, tambahkan metadata ke pengguna dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya, lihat Tag untuk AWS Identity and Access Management sumber daya.

7. Verifikasi keanggotaan grup pengguna untuk pengguna baru. Saat Anda siap untuk melanjutkan, pilih Create user (Buat pengguna).

8. Pemberitahuan status muncul memberi tahu Anda bahwa pengguna berhasil dibuat. Pilih Lihat pengguna untuk membuka halaman detail pengguna

9. Pilih tab Security credentials. Kemudian buat kredensional yang diperlukan untuk beban kerja.

   - Kunci akses —Pilih Buat kunci akses untuk menghasilkan dan mengunduh kunci akses untuk pengguna.

> **Important**
>
> Ini adalah satu-satunya kesempatan Anda untuk melihat atau mengunduh kunci akses rahasia, dan Anda harus memberikan informasi ini kepada pengguna Anda sebelum mereka dapat menggunakan AWS API. Simpan access key ID baru

pengguna dan secret access key di tempat yang aman dan terlindungi. Anda tidak akan memiliki akses ke kunci rahasia lagi setelah langkah ini.

- SSHkunci publik untuk AWS CodeCommit —Pilih Unggah kunci SSH publik untuk mengunggah kunci SSH publik sehingga pengguna dapat berkomunikasi dengan CodeCommit repositori. SSH
- HTTPSGit credentials for AWS CodeCommit —Select Generate credentials untuk menghasilkan kumpulan kredensi pengguna yang unik untuk digunakan dengan repositori Git. Pilih Unduh kredensi untuk menyimpan nama pengguna dan kata sandi ke file.csv. Ini adalah satu-satunya waktu informasi tersedia. Jika Anda lupa atau kehilangan kata sandi, Anda harus mengatur ulang kata sandi.
- Kredensial untuk Amazon Keyspaces (untuk Apache Cassandra) —Pilih Hasilkan kredensi untuk menghasilkan kredensi pengguna khusus layanan untuk digunakan dengan Amazon Keyspaces. Pilih Unduh kredensi untuk menyimpan nama pengguna dan kata sandi ke file.csv. Ini adalah satu-satunya waktu informasi tersedia. Jika Anda lupa atau kehilangan kata sandi, Anda harus mengatur ulang kata sandi.

  Important

  Kredensi khusus layanan adalah kredensi jangka panjang yang terkait dengan IAM pengguna tertentu dan hanya dapat digunakan untuk layanan yang mereka buat. Untuk memberikan izin IAM peran atau identitas gabungan untuk mengakses semua AWS sumber daya Anda menggunakan kredensi sementara, gunakan AWS autentikasi dengan plugin otentikasi SiGv4 untuk Amazon Keyspaces. Untuk informasi selengkapnya lihat, Menggunakan kredensi sementara untuk terhubung ke Amazon Keyspaces (untuk Apache Cassandra) menggunakan IAM peran dan plugin SiGv4 di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

- Sertifikat penandatanganan X.509 —Pilih Buat Sertifikat X.509 jika Anda perlu membuat permintaan SOAP protokol yang aman dan berada di Wilayah yang tidak didukung oleh. AWS Certificate Manager ACMadalah alat yang disukai untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Untuk informasi selengkapnya tentang penggunaanACM, lihat Panduan AWS Certificate Manager Pengguna.

Anda telah membuat pengguna dengan akses terprogram dan mengonfigurasinya dengan fungsi PowerUserAccesspekerjaan. Kebijakan izin pengguna ini memberikan akses penuh ke setiap layanan kecuali untuk IAM dan. AWS Organizations

Anda dapat menggunakan proses yang sama ini untuk memberikan beban kerja tambahan akses terprogram ke Akun AWS sumber daya Anda, jika beban kerja tidak dapat mengambil peran. IAM Prosedur ini menggunakan kebijakan PowerUserAccessterkelola untuk menetapkan izin. Untuk mengikuti praktik terbaik dengan hak istimewa terkecil, pertimbangkan untuk menggunakan kebijakan yang lebih ketat atau membuat kebijakan khusus yang membatasi akses hanya ke sumber daya yang diperlukan oleh program. Untuk mempelajari cara menggunakan kebijakan yang membatasi izin pengguna ke AWS sumber daya tertentu, lihat Manajemen akses untuk AWS sumber daya dan. Contoh kebijakan berbasis identitas IAM Untuk menambahkan pengguna tambahan ke grup pengguna setelah dibuat, lihat Mengedit pengguna dalam IAM IAM grup.

AWS CLI

1. Buat pengguna bernama**`Automation`**.

   - aws iam buat-pengguna

   ```
   aws iam create-user \
       --user-name Automation
   ```

2. Buat grup IAM pengguna bernama**`AutomationGroup`**, lampirkan kebijakan AWS terkelola `PowerUserAccess` ke grup, lalu tambahkan **`Automation`** pengguna ke grup.

   > **Note**
   >
   > Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Setiap kebijakan memiliki Nama Sumber Daya Amazon (ARN) sendiri yang menyertakan nama kebijakan. Misalnya, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` adalah kebijakan yang AWS dikelola. Untuk informasi lebih lanjut tentangARNs, lihatIAM ARNs. Untuk mengetahui daftar kebijakan AWS terkelola Layanan AWS, lihat kebijakan AWS terkelola.

   - aws iam membuat grup

```
aws iam create-group \
    --group-name AutomationGroup
```

- aws iam attach-group-policy

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/PowerUserAccess \
    --group-name AutomationGroup
```

- aws iam add-user-to -grup

```
aws iam add-user-to-group \
    --user-name Automation \
    --group-name AutomationGroup
```

- Jalankan perintah aws iam get-group untuk mencantumkan **AutomationGroup** dan anggotanya.

```
aws iam get-group \
    --group-name AutomationGroup
```

3. Buat kredenal keamanan yang diperlukan untuk beban kerja.

   - Buat kunci akses untuk pengujian — aws iam create-access-key

```
aws iam create-access-key \
    --user-name Automation
```

   Output dari perintah ini menampilkan kunci akses rahasia dan ID kunci akses. Rekam dan simpan informasi ini di lokasi yang aman. Jika kredensi ini hilang, mereka tidak dapat dipulihkan, dan Anda harus membuat kunci akses baru.

> **Important**
>
> Kunci akses IAM pengguna ini adalah kredensi jangka panjang yang menghadirkan risiko keamanan ke akun Anda. Setelah Anda menyelesaikan pengujian, kami sarankan Anda menghapus kunci akses ini. Jika Anda memiliki skenario di mana Anda mempertimbangkan kunci akses, selidiki apakah Anda dapat mengaktifkan MFA IAM pengguna beban kerja Anda dan gunakan aws sts get-session-token untuk mendapatkan kredensi sementara untuk sesi tersebut alih-alih menggunakan IAM kunci akses.

- Unggah kunci SSH publik untuk AWS CodeCommit — aws iam upload-ssh-public -key

  Contoh berikut mengasumsikan bahwa Anda memiliki kunci SSH publik yang disimpan dalam file`sshkey.pub`.

  ```
  aws upload-ssh-public-key \
      --user-name Automation \
      --ssh-public-key-body file://sshkey.pub
  ```

- Unggah sertifikat penandatanganan X.509 — aws iam upload-signing-certificate

  Unggah sertifikat X.509 jika Anda perlu membuat permintaan SOAP protokol aman dan berada di Wilayah yang tidak didukung oleh. AWS Certificate Manager ACMadalah alat yang disukai untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Untuk informasi selengkapnya tentang penggunaanACM, lihat Panduan AWS Certificate Manager Pengguna.

  Contoh berikut mengasumsikan bahwa Anda memiliki sertifikat penandatanganan X.509 yang disimpan dalam file. `certificate.pem`

  ```
  aws iam upload-signing-certificate \
  --user-name Automation \
  --certificate-body file://certificate.pem
  ```

Anda dapat menggunakan proses yang sama ini untuk memberikan beban kerja tambahan akses terprogram ke Akun AWS sumber daya Anda, jika beban kerja tidak dapat mengambil peran. IAM Prosedur ini menggunakan kebijakan PowerUserAccessterkelola untuk menetapkan izin. Untuk mengikuti praktik terbaik dengan hak istimewa terkecil, pertimbangkan untuk menggunakan kebijakan yang lebih ketat atau membuat kebijakan khusus yang membatasi akses hanya ke sumber daya yang diperlukan oleh program. Untuk mempelajari cara menggunakan kebijakan yang membatasi izin pengguna ke AWS sumber daya tertentu, lihat Manajemen akses untuk AWS sumber daya dan. Contoh kebijakan berbasis identitas IAM Untuk menambahkan pengguna tambahan ke grup pengguna setelah dibuat, lihat Mengedit pengguna dalam IAM IAM grup.

Tambahkan otentikasi multi-faktor untuk identitas Anda

Menambahkan otentikasi multi-faktor (MFA) untuk identitas Anda adalah rekomendasi praktik terbaik lainnya. MFAadalah lapisan keamanan tambahan yang mengharuskan pengguna untuk memberikan faktor otentikasi tambahan setelah memberikan nama pengguna dan kata sandi mereka untuk memverifikasi identitas mereka. Ini secara signifikan meningkatkan keamanan dengan membuatnya lebih sulit bagi penyerang untuk mendapatkan akses yang tidak sah, bahkan jika kata sandi pengguna dikompromikan. MFAsecara luas diadopsi sebagai praktik terbaik untuk mengamankan akses ke akun online, layanan cloud, dan sumber daya sensitif lainnya. AWS dukungan MFA untuk pengguna root, pengguna, IAM pengguna di IAM Identity Center, Builder ID, dan pengguna federasi. Untuk keamanan tambahan, Anda dapat membuat kebijakan yang diperlukan MFA sebelum mengizinkan pengguna mengakses sumber daya atau mengambil tindakan tertentu dan melampirkan kebijakan ini ke IAM peran Anda.

Untuk informasi selengkapnya, lihat Mengaktifkan MFA di Pusat IAM Identitas danAWS Otentikasi multi-faktor di IAM.

Bersiaplah untuk izin hak istimewa paling sedikit

Menggunakan izin hak istimewa paling rendah adalah rekomendasi praktik terbaik. IAM Konsep izin hak istimewa paling rendah adalah memberi pengguna izin yang diperlukan untuk melakukan tugas dan tidak ada izin tambahan. Saat Anda menyiapkan, pertimbangkan bagaimana Anda akan mendukung izin hak istimewa paling sedikit. Pengguna root, pengguna administratif, dan IAM pengguna akses darurat memiliki izin kuat yang tidak diperlukan untuk tugas sehari-hari. Saat Anda mempelajari AWS dan menguji berbagai layanan, kami menyarankan Anda membuat setidaknya satu pengguna tambahan di Pusat IAM Identitas dengan izin yang lebih rendah yang dapat Anda gunakan dalam skenario yang berbeda. Anda dapat menggunakan IAM kebijakan untuk menentukan tindakan

yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu dan kemudian terhubung ke sumber daya tersebut dengan akun istimewa Anda yang lebih rendah.

Jika Anda menggunakan Pusat IAM Identitas, pertimbangkan untuk menggunakan set izin Pusat IAM Identitas untuk memulai. Untuk mempelajari selengkapnya, lihat Membuat set izin di Panduan Pengguna Pusat IAM Identitas.

Jika Anda tidak menggunakan Pusat IAM Identitas, gunakan IAM peran untuk menentukan izin untuk IAM entitas yang berbeda. Untuk mempelajari selengkapnya, lihat IAMpenciptaan peran.

Kedua IAM peran dan set izin Pusat IAM Identitas dapat menggunakan kebijakan AWS terkelola berdasarkan fungsi pekerjaan. Untuk detail tentang izin yang diberikan oleh kebijakan ini, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

> **Important**
>
> Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan oleh semua pelanggan. AWS Setelah disiapkan, sebaiknya gunakan IAM Access Analyzer untuk membuat kebijakan hak istimewa paling sedikit berdasarkan aktivitas akses yang masuk. AWS CloudTrail Untuk informasi selengkapnya tentang pembuatan kebijakan, lihat pembuatan kebijakan IAM Access Analyzer.

Saat memulai, sebaiknya gunakan kebijakan AWS terkelola untuk memberikan izin. Setelah periode aktivitas sampel yang telah ditentukan sebelumnya (seperti 90 hari) berlalu, Anda dapat meninjau layanan yang diakses orang dan beban kerja. Kemudian Anda dapat membuat kebijakan terkelola pelanggan baru dengan izin yang dikurangi untuk mengganti kebijakan AWS terkelola. Kebijakan baru harus mencakup hanya layanan yang diakses selama periode sampel. Perbarui izin Anda untuk menghapus kebijakan AWS terkelola dan melampirkan kebijakan terkelola pelanggan baru yang Anda buat.

Meninjau informasi terakhir yang diakses untuk akun Anda AWS

Anda dapat melihat layanan informasi yang terakhir diakses untuk IAM menggunakan IAM konsol, AWS CLI, atau AWS API. Untuk informasi penting tentang data, izin yan diperlukan, pemecahan masalah, dan region yang didukung lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Anda dapat melihat informasi untuk jenis sumber daya berikut diIAM. Dalam setiap kejadian, informasi tersebut mencakup layanan yang diizinkan untuk periode pelaporan tertentu:

- IAMpengguna — Lihat terakhir kali pengguna mencoba mengakses setiap layanan yang diizinkan.
- IAMgrup — Melihat informasi tentang terakhir kali anggota IAM grup mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total anggota yang mencoba mengaksesnya.
- IAMperan — Lihat terakhir kali seseorang menggunakan peran tersebut dalam upaya mengakses setiap layanan yang diizinkan.
- Kebijakan – Melihat informasi terakhir kali pengguna atau peran mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total entitas yang mencoba mengaksesnya.

> **Note**
>
> Sebelum Anda melihat informasi akses untuk sumber dayaIAM, pastikan Anda memahami periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi untuk informasi Anda. Untuk detail selengkapnya, lihat the section called “Hal yang perlu diketahui tentang informasi yang terakhir diakses”.

Untuk informasi selengkapnya tentang informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Untuk meninjau informasi yang terakhir diakses untuk Akun AWS

Pilih tab untuk metode yang ingin Anda ikuti untuk meninjau informasi yang terakhir diakses:

IAM console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Di halaman Beranda Konsol, pilih IAM layanan.
3. Di panel navigasi, pilih Grup pengguna, Pengguna, Peran, atau Kebijakan.
4. Pilih nama pengguna, grup pengguna, peran, atau kebijakan apa pun untuk membuka halaman Ringkasannya dan pilih tab Terakhir Diakses. Lihat informasi berikut, berdasarkan sumber daya yang Anda pilih:

- Grup pengguna - Lihat daftar layanan yang dapat diakses oleh anggota grup pengguna. Anda juga dapat melihat kapan anggota terakhir mengakses layanan, kebijakan grup pengguna apa yang mereka gunakan, dan anggota grup pengguna mana yang membuat permintaan. Pilih nama kebijakan untuk dipelajari baik merupakan kebijakan terkelola atau kebijakan grup pengguna inline. Pilih nama anggota grup pengguna untuk melihat semua anggota grup pengguna dan kapan mereka terakhir kali mengakses layanan.
- Pengguna – Lihat daftar layanan yang dapat diakses pengguna. Anda juga dapat melihat kapan mereka terakhir mengakses layanan, dan kebijakan apa yang saat ini terkait dengan pengguna. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola, kebijakan pengguna inline, atau kebijakan inline untuk grup pengguna.
- Peran – Lihat daftar layanan yang dapat diakses oleh peran, kapan peran terakhir mengakses layanan, dan kebijakan apa yang digunakan. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola atau kebijakan peran selaras.
- Kebijakan – Lihat daftar layanan dengan tindakan yang diperbolehkan dalam kebijakan. Anda juga dapat melihat kapan kebijakan terakhir digunakan untuk mengakses layanan, dan entitas (pengguna atau peran) mana yang menggunakan kebijakan tersebut. Tanggal Terakhir diakses juga mencakup kapan akses diberikan ke kebijakan ini melalui kebijakan lain. Pilih nama entitas untuk dipelajari yang sudah terlampirkan dengan kebijakan ini dan kapan terakhir kali mereka mengakses layanan.

5. Di kolom Layanan tabel, pilih nama salah satu layanan yang menyertakan informasi tindakan yang terakhir diakses untuk melihat daftar tindakan manajemen yang coba diakses oleh IAM entitas. Anda dapat melihat Wilayah AWS dan stempel waktu yang menunjukkan kapan seseorang terakhir mencoba melakukan tindakan.
6. Kolom Terakhir diakses ditampilkan untuk layanan dan tindakan manajemen layanan yang menyertakan informasi tindakan yang terakhir diakses. Tinjau hasil-hasil berikut yang kemungkinan muncul dalam kolom ini. Hasil ini bervariasi tergantung pada apakah layanan atau tindakan diizinkan, diakses, dan apakah terlacak oleh AWS untuk informasi yang terakhir diakses.

   <jumlah> hari yang lalu

   Jumlah hari sejak layanan atau tindakan digunakan dalam periode pelacakan. Periode pelacakan layanan adalah selama 400 hari terakhir. Periode pelacakan untuk tindakan Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk Amazon EC2IAM,, dan tindakan Lambda dimulai pada 7 April 2021. Periode pelacakan untuk semua layanan

lainnya dimulai pada 23 Mei 2023. Untuk mempelajari selengkapnya tentang melacak tanggal mulai masing-masing Wilayah AWS, lihatTempat AWS melacak informasi terakhir yang diakses.

Tidak diakses dalam periode pelacakan

Layanan atau tindakan yang dilacak belum digunakan oleh entitas dalam periode pelacakan.

Anda dapat memiliki izin untuk tindakan yang tidak muncul dalam daftar. Ini dapat terjadi jika informasi pelacakan untuk tindakan saat ini tidak disertakan oleh AWS. Anda tidak diperkenankan mengambil keputusan izin hanya berdasarkan ketiadaan informasi pelacakan. Alih-alih, kami menyarankan agar Anda menggunakan informasi ini untuk menginformasikan dan mendukung strategi Anda secara keseluruhan untuk memberikan privilese paling sedikit. Periksa kebijakan Anda untuk mengonfirmasi bahwa tingkat akses sesuai.

AWS CLI

Anda dapat menggunakan AWS CLI untuk mengambil informasi tentang terakhir kali IAM sumber daya di Anda Akun AWS digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, EC2 AmazonIAM, dan Lambda. IAMSumber daya dapat berupa pengguna, grup pengguna, peran, atau kebijakan.

- Hasilkan laporan untuk IAM sumber daya di file Akun AWS. Permintaan harus menyertakan IAM sumber daya (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. ARN Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat detail akses baik untuk layanan maupun layanan beserta tindakan. Permintaan tersebut menghasilkan `job-id` yang kemudian dapat Anda gunakan di operasi `get-service-last-accessed-details` dan `get-service-last-accessed-details-with-entities` untuk memonitor `job-status` hingga pekerjaan selesai.

  - aws iam generate-service-last -detail yang dapat diakses

  a. Dapatkan detail laporan menggunakan parameter `job-id` dari langkah sebelumnya.

     - aws iam get-service-last -detail yang dapat diakses

Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `generate-service-last-accessed-details` operasi:

- Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir pengguna dan pengguna. ARN
- Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ini juga mengembalikan ARN pengguna itu dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan. Gunakan GetServiceLastAccessedDetailsWithEntitiesoperasi untuk mengambil daftar semua anggota.
- Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir peran dan peran. ARN
- Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna atau peran) untuk mengakses layanan menggunakan kebijakan. Ini juga mengembalikan ARN entitas itu dan jumlah total entitas yang mencoba mengakses.

b. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini mengembalikan daftar entitas dengan masing-masing entitasARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - aws iam get-service-last - accessed-details-with-entities

c. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti

kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas IAM izin, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Mengevaluasi kebijakan dalam satu akun.

- aws iam list-policies-granting -layanan-akses

API

Anda dapat menggunakan AWS API untuk mengambil informasi tentang terakhir kali IAM sumber daya digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, EC2 AmazonIAM, dan Lambda. IAMSumber daya dapat berupa pengguna, grup pengguna, peran, atau kebijakan. Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat baik layanan maupun layanan beserta tindakan.

1. Buat laporan. Permintaan harus menyertakan IAM sumber daya (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. ARN Ia menghasilkan `JobId` yang kemudian dapat Anda gunakan di operasi `GetServiceLastAccessedDetails` dan `GetServiceLastAccessedDetailsWithEntities` untuk memonitor `JobStatus` hingga pekerjaan selesai.

   - GenerateServiceLastAccessedDetails

2. Dapatkan detail laporan menggunakan parameter `JobId` dari langkah sebelumnya.

   - GetServiceLastAccessedDetails

   Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `GenerateServiceLastAccessedDetails` operasi:

   - Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir pengguna dan pengguna. ARN
   - Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ini juga mengembalikan ARN pengguna itu dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan.

Gunakan GetServiceLastAccessedDetailsWithEntitiesoperasi untuk mengambil daftar semua anggota.

- Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir peran dan peran. ARN
- Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna atau peran) untuk mengakses layanan menggunakan kebijakan. Ini juga mengembalikan ARN entitas itu dan jumlah total entitas yang mencoba mengakses.

3. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini mengembalikan daftar entitas dengan masing-masing entitasARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - GetServiceLastAccessedDetailsWithEntities

4. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas IAM izin, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Mengevaluasi kebijakan dalam satu akun.

   - ListPoliciesGrantingServiceAccess

Menghasilkan kebijakan berdasarkan aktivitas akses

Anda dapat menggunakan aktivitas akses yang direkam AWS CloudTrail untuk IAM pengguna atau IAM peran agar IAM Access Analyzer menghasilkan kebijakan terkelola pelanggan untuk mengizinkan akses hanya ke layanan yang dibutuhkan pengguna dan peran tertentu.

Saat IAM Access Analyzer membuat IAM kebijakan, informasi akan dikembalikan untuk membantu Anda menyesuaikan kebijakan lebih lanjut. Dua kategori informasi dapat dihasilkan ketika kebijakan yang membuat:

- Kebijakan dengan informasi tingkat tindakan — Untuk beberapa AWS layanan, seperti AmazonEC2, IAM Access Analyzer dapat mengidentifikasi tindakan yang ditemukan dalam CloudTrail acara Anda dan mencantumkan tindakan yang digunakan dalam kebijakan yang dihasilkannya. Untuk daftar layanan yang didukung, lihatIAMLayanan pembuatan kebijakan Access Analyzer. Untuk beberapa layanan, IAM Access Analyzer meminta Anda untuk menambahkan tindakan untuk layanan ke kebijakan yang dihasilkan.
- Kebijakan dengan informasi tingkat layanan — IAM Access Analyzer menggunakan informasi yang terakhir diakses untuk membuat templat kebijakan dengan semua layanan yang baru digunakan. Saat menggunakan AWS Management Console, kami meminta Anda untuk meninjau layanan dan menambahkan tindakan untuk menyelesaikan kebijakan.

Untuk menghasilkan kebijakan berdasarkan aktivitas akses

Dalam prosedur berikut, kami akan mengurangi izin yang diberikan ke peran agar sesuai dengan penggunaan pengguna. Saat Anda memilih pengguna, pilih pengguna yang penggunaannya mencontohkan peran tersebut. Banyak pelanggan menyiapkan akun pengguna uji dengan PowerUserizin dan kemudian meminta mereka melakukan serangkaian tugas tertentu untuk periode waktu yang singkat untuk menentukan akses apa yang diperlukan untuk melakukan tugas-tugas tersebut,

Pilih tab untuk metode yang ingin Anda ikuti untuk menghasilkan kebijakan izin baru:

IAM console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Di halaman Beranda Konsol, pilih IAM layanan.
3. Di panel navigasi, pilih Pengguna dan kemudian pilih nama pengguna untuk masuk ke halaman detail pengguna.
4. Pada tab Izin, di bawah Buat kebijakan berdasarkan CloudTrail peristiwa, pilih Buat kebijakan.
5. Pada halaman Buat kebijakan, konfigurasikan item berikut:
   - Untuk Pilih periode waktu, pilih 7 hari terakhir.
   - Untuk CloudTrail jejak yang akan dianalisis, pilih Wilayah dan jejak tempat aktivitas pengguna ini direkam.

- Pilih Buat dan gunakan peran layanan baru.

6. Pilih Buat kebijakan lalu tunggu hingga peran dibuat. Jangan me-refresh atau menjauh dari halaman konsol hingga muncul pesan notifikasi pembuatan Kebijakan yang sedang berlangsung.
7. Setelah kebijakan dibuat, Anda harus meninjau dan menyesuaikannya sesuai kebutuhan dengan akun IDs dan ARNs sumber daya. Selain itu, kebijakan yang dibuat secara otomatis mungkin tidak menyertakan informasi tingkat tindakan yang diperlukan untuk menyelesaikan kebijakan. Untuk informasi selengkapnya lihat, pembuatan kebijakan IAM Access Analyzer.

   Misalnya, Anda dapat mengedit pernyataan pertama yang menyertakan `Allow` efek dan `NotAction` elemen untuk mengizinkan hanya tindakan Amazon EC2 dan Amazon S3. Untuk melakukan ini, ganti dengan pernyataan yang memiliki ID `FullAccessToSomeServices`. Kebijakan baru Anda bisa terlihat seperti contoh kebijakan berikut.

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
         {
             "Sid": "FullAccessToSomeServices",
             "Effect": "Allow",
             "Action": [
                 "ec2:*",
                 "s3:*"
             ],
             "Resource": "*"
         },
         {
             "Effect": "Allow",
             "Action": [
                 "iam:CreateServiceLinkedRole",
                 "iam:DeleteServiceLinkedRole",
                 "iam:ListRoles",
                 "organizations:DescribeOrganization"
             ],
             "Resource": "*"
         }
     ]
   }
   ```

8. Untuk mendukung praktik terbaik memberikan hak istimewa paling rendah, tinjau dan perbaiki kesalahan, peringatan, atau saran yang dikembalikan selama validasi kebijakan.

9. Untuk mengurangi izin kebijakan Anda terhadap tindakan dan sumber daya tertentu, lihat acara Anda di CloudTrail riwayat Acara. Di sana Anda dapat melihat informasi terperinci tentang tindakan dan sumber daya spesifik yang telah diakses oleh pengguna Anda. Untuk informasi selengkapnya, lihat Melihat CloudTrail Acara di CloudTrail Konsol di Panduan AWS CloudTrail Pengguna.
10. Setelah meninjau dan memvalidasi kebijakan Anda, simpan dengan nama deskriptif.
11. Arahkan ke halaman Peran dan pilih peran yang akan diambil orang saat mereka melakukan tugas yang diizinkan oleh kebijakan baru Anda.
12. Pilih tab Izin, lalu pilih Tambahkan izin dan pilih Lampirkan kebijakan.
13. Pada halaman Lampirkan kebijakan izin, di daftar Kebijakan izin lainnya, pilih kebijakan yang Anda buat, lalu pilih Lampirkan kebijakan.
14. Anda dikembalikan ke halaman Detail peran. ada dua kebijakan yang dilampirkan peran, kebijakan AWS terkelola sebelumnya, seperti PowerUserAccess, dan kebijakan baru Anda. Pilih kotak centang untuk kebijakan AWS terkelola, lalu pilih Hapus. Ketika diminta untuk mengonfirmasi penghapusan, pilih Hapus.

IAMpengguna, pengguna gabungan, dan beban kerja yang mengambil peran ini sekarang telah mengurangi akses sesuai dengan kebijakan baru yang Anda buat.

AWS CLI

Anda dapat menggunakan perintah berikut untuk menghasilkan kebijakan menggunakan AWS CLI.

Untuk membuat kebijakan

- aws accessanalyzer start-policy-generation

Untuk melihat kebijakan yang dibuat

- aws accessanalyzer get-generated-policy

Untuk membatalkan permintaan pembuatan kebijakan

- aws accessanalyzer cancel-policy-generation

Untuk melihat daftar permintaan pembuatan kebijakan

- aws accessanalyzer list-policy-generations

API

Anda dapat menggunakan operasi berikut untuk membuat kebijakan menggunakan AWS API.

Untuk membuat kebijakan

- StartPolicyGeneration

Untuk melihat kebijakan yang dibuat

- GetGeneratedPolicy

Untuk membatalkan permintaan pembuatan kebijakan

- CancelPolicyGeneration

Untuk melihat daftar permintaan pembuatan kebijakan

- ListPolicyGenerations

Menggunakan pencarian untuk menemukan IAM sumber daya

Saat Anda mengerjakan temuan akses, Anda dapat menggunakan halaman pencarian IAM konsol sebagai opsi yang lebih cepat untuk menemukan IAM sumber daya. Anda dapat mencari sumber daya menggunakan nama sumber daya sebagian atauARNs.

IAM console

Fitur pencarian IAM konsol dapat menemukan salah satu dari berikut ini:

- IAMnama entitas yang cocok dengan kata kunci penelusuran Anda (untuk pengguna, grup, peran, penyedia identitas, dan kebijakan)
- Tugas yang cocok dengan kata kunci pencarian Anda

Fitur pencarian IAM konsol tidak menampilkan informasi tentang IAM Access Analyzer.

Setiap baris dalam hasil pencarian adalah tautan aktif. Misalnya, Anda dapat memilih nama pengguna di hasil pencarian, yang membawa Anda ke laman detail pengguna tersebut. Atau Anda dapat memilih tautan tindakan, misalnya Buat pengguna, untuk menuju ke laman Buat Pengguna.

> **Note**
>
> Pencarian access key mengharuskan Anda mengetikkan access key ID penuh di kotak pencarian. Hasil pencarian menampilkan pengguna yang terkait dengan kunci tersebut. Dari sana Anda dapat menavigasi langsung ke halaman pengguna itu, di mana Anda dapat mengelola kunci akses.

Gunakan halaman Pencarian di IAM konsol untuk menemukan item yang terkait dengan akun itu.

Untuk mencari item di IAM konsol

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Di halaman Beranda Konsol, pilih IAM layanan.
3. Di panel navigasi, pilih Cari.
4. Di kotak Cari ketikkan kata kunci pencarian Anda.
5. Pilih tautan di daftar hasil pencarian untuk menavigasi ke bagian konsol yang sesuai.

Ikon berikut mengidentifikasi jenis-jenis IT yang ditemukan dengan pencarian:

Ikon	Deskripsi
	IAMpengguna
	IAMkelompok
	IAMperan

Ikon	Deskripsi
	IAMkebijakan
	Tugas seperti “buat pengguna” atau “lampirkan kebijakan”
	Hasil dari kata kunci **`delete`**

Contoh frase pencarian

Anda dapat menggunakan frasa berikut dalam IAM pencarian. Ganti istilah dalam huruf miring dengan nama IAM pengguna, grup, peran, kunci akses, kebijakan, atau penyedia identitas aktual yang ingin Anda temukan.

- ***`user_name`*** atau ***`group_name`*** atau ***`role_name`*** atau ***`policy_name`*** atau ***`identity_provider_name`***
- ***`access_key`***
- **`add user`** ***`user_name`*** **`to groups`** atau **`add users to group`** ***`group_name`***
- **`remove user`** ***`user_name`*** **`from groups`**
- **`delete`** ***`user_name`*** atau **`delete`** ***`group_name`*** atau **`delete`** ***`role_name`***, atau **`delete`** ***`policy_name`***, atau **`delete`** ***`identity_provider_name`***
- **`manage access keys`** ***`user_name`***
- **`manage signing certificates`** ***`user_name`***
- **`users`**
- **`manage MFA for`** ***`user_name`***
- **`manage password for`** ***`user_name`***
- **`create role`**
- **`password policy`**
- **`edit trust policy for role`** ***`role_name`***
- **`show policy document for role`** ***`role_name`***
- **`attach policy to`** ***`role_name`***

- **create managed policy**
- **create user**
- **create group**
- **attach policy to *group_name***
- **attach entities to *policy_name***
- **detach entities from *policy_name***

Praktik terbaik keamanan dan kasus penggunaan AWS Identity and Access Management

AWS Identity and Access Management (IAM) menawarkan beberapa fitur keamanan untuk dipertimbangkan saat Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap.

Praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan spesifik Anda. Oleh karena itu, perlakukan mereka sebagai pertimbangan yang bermanfaat daripada persyaratan.

Untuk mendapatkan hasil maksimalIAM, luangkan waktu untuk mempelajari praktik terbaik yang direkomendasikan. Salah satu cara efektif untuk melakukan ini adalah dengan melihat bagaimana IAM digunakan dalam skenario dunia nyata untuk bekerja dengan yang lain Layanan AWS.

Topik

Praktik terbaik keamanan di IAM

Follow us on Twitter

Untuk membantu mengamankan AWS sumber daya Anda, ikuti praktik terbaik ini untuk AWS Identity and Access Management (IAM).

Topik

- Terapkan izin hak istimewa paling sedikit
- Memulai kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit
- Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa paling sedikit berdasarkan aktivitas akses
- Tinjau dan hapus pengguna, peran, izin, kebijakan, dan kredensil yang tidak digunakan secara teratur
- Gunakan ketentuan dalam IAM kebijakan untuk membatasi akses lebih lanjut
- Verifikasi akses publik dan lintas akun ke sumber daya dengan IAM Access Analyzer
- Gunakan IAM Access Analyzer untuk memvalidasi IAM kebijakan Anda guna memastikan izin yang aman dan fungsional
- Menetapkan pagar pembatas izin di beberapa akun
- Gunakan batas izin untuk mendelegasikan manajemen izin dalam akun

Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara

Pengguna manusia, juga dikenal sebagai identitas manusia, adalah orang, administrator, pengembang, operator, dan konsumen aplikasi Anda. Mereka harus memiliki identitas untuk mengakses AWS lingkungan dan aplikasi Anda. Pengguna manusia yang merupakan anggota organisasi Anda juga dikenal sebagai identitas tenaga kerja. Pengguna manusia juga dapat menjadi pengguna eksternal dengan siapa Anda berkolaborasi, dan yang berinteraksi dengan AWS sumber daya Anda. Mereka dapat melakukan ini melalui browser web, aplikasi klien, aplikasi seluler, atau alat baris perintah interaktif.

Minta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Anda dapat menggunakan penyedia identitas bagi pengguna manusia Anda untuk menyediakan akses federasi Akun AWS dengan mengambil peran, yang menyediakan kredensi sementara. Untuk manajemen akses terpusat, kami menyarankan Anda menggunakan AWS IAM Identity Center (Pusat IAM Identitas) untuk mengelola akses ke akun Anda dan izin dalam akun tersebut. Anda dapat mengelola identitas pengguna dengan Pusat IAM Identitas, atau mengelola izin akses untuk identitas pengguna di Pusat Identitas dari penyedia IAM identitas eksternal. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center dalam Panduan Pengguna AWS IAM Identity Center .

Untuk informasi lebih lanjut tentang peran, lihat Istilah dan konsep peran.

Memerlukan beban kerja untuk menggunakan kredensil sementara dengan peran untuk mengakses IAM AWS

Beban kerja adalah kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi atau proses backend. Beban kerja Anda dapat memiliki aplikasi, alat operasional, dan komponen yang memerlukan identitas untuk membuat permintaan Layanan AWS, seperti permintaan untuk membaca data. Identitas ini mencakup mesin yang berjalan di AWS lingkungan Anda, seperti EC2 instans atau AWS Lambda fungsi Amazon.

Anda juga dapat mengelola identitas mesin untuk pihak eksternal yang membutuhkan akses. Untuk memberikan akses ke identitas mesin, Anda dapat menggunakan IAM peran. IAMperan memiliki izin khusus dan menyediakan cara untuk mengakses AWS dengan mengandalkan kredenal keamanan sementara dengan sesi peran. Selain itu, Anda mungkin memiliki mesin di luar AWS yang membutuhkan akses ke AWS lingkungan Anda. Untuk mesin yang berjalan di luar AWS Anda dapat menggunakan AWS Identity and Access Management Peran Di Mana Saja. Untuk informasi lebih lanjut tentang peran, lihat IAMperan. Untuk detail tentang cara menggunakan peran untuk mendelegasikan akses di seluruh Akun AWS, lihatIAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran.

Memerlukan otentikasi multi-faktor () MFA

Sebaiknya gunakan IAM peran untuk pengguna manusia dan beban kerja yang mengakses AWS sumber daya Anda sehingga mereka menggunakan kredensi sementara. Namun, untuk skenario di mana Anda memerlukan IAM pengguna atau pengguna root di akun Anda, memerlukan MFA keamanan tambahan. DenganMFA, pengguna memiliki perangkat yang menghasilkan respons terhadap tantangan otentikasi. Setiap kredensi pengguna dan respons yang dihasilkan perangkat diperlukan untuk menyelesaikan proses masuk. Untuk informasi selengkapnya, lihat AWS Otentikasi multi-faktor di IAM.

Jika Anda menggunakan Pusat IAM Identitas untuk manajemen akses terpusat untuk pengguna manusia, Anda dapat menggunakan MFA kemampuan Pusat IAM Identitas ketika sumber identitas Anda dikonfigurasi dengan toko IAM identitas Pusat Identitas, Microsoft AD AWS Terkelola, atau AD Connector. Untuk informasi selengkapnya tentang MFA di Pusat IAM Identitas, lihat Autentikasi multi-faktor di AWS IAM Identity Center Panduan Pengguna.

Perbarui kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredensi jangka panjang

Jika memungkinkan, kami sarankan untuk mengandalkan kredensi sementara alih-alih membuat kredensi jangka panjang seperti kunci akses. Namun, untuk skenario di mana Anda memerlukan IAM pengguna dengan akses terprogram dan kredensi jangka panjang, kami sarankan Anda memperbarui kunci akses saat diperlukan, seperti ketika seorang karyawan meninggalkan perusahaan Anda. Kami menyarankan Anda menggunakan IAMakses informasi yang terakhir digunakan untuk memperbarui dan menghapus kunci akses dengan aman. Untuk informasi selengkapnya, lihat Perbarui kunci akses.

Ada kasus penggunaan khusus yang memerlukan kredensil jangka panjang dengan IAM pengguna di. AWS Beberapa kasus penggunaan meliputi:

- Beban kerja yang tidak dapat menggunakan IAM peran — Anda dapat menjalankan beban kerja dari lokasi yang perlu diakses. AWS Dalam beberapa situasi, Anda tidak dapat menggunakan IAM peran untuk memberikan kredensi sementara, seperti untuk WordPress plugin. Dalam situasi ini, gunakan kunci akses jangka panjang IAM pengguna untuk beban kerja tersebut untuk mengautentikasi. AWS
- AWS Klien pihak ketiga — Jika Anda menggunakan alat yang tidak mendukung akses dengan Pusat IAM Identitas, seperti AWS klien pihak ketiga atau vendor yang tidak di-host AWS, gunakan kunci akses jangka panjang IAM pengguna.
- AWS CodeCommit akses - Jika Anda menggunakan CodeCommit untuk menyimpan kode Anda, Anda dapat menggunakan IAM pengguna dengan SSH kunci atau kredensi khusus layanan untuk mengautentikasi CodeCommit ke repositori Anda. Kami menyarankan Anda melakukan ini selain menggunakan pengguna di Pusat IAM Identitas untuk otentikasi normal. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud Anda Akun AWS atau ke aplikasi cloud Anda. Untuk memberi pengguna akses ke CodeCommit repositori Anda tanpa mengonfigurasi IAM pengguna, Anda dapat mengonfigurasi utilitas. git-remote-codecommit Untuk informasi lebih lanjut tentang IAM dan CodeCommit, lihatIAMkredensial untuk: Kredensial CodeCommit Git, SSH kunci, dan kunci akses AWS. Untuk informasi selengkapnya tentang mengonfigurasi git-remote-codecommit utilitas, lihat Menghubungkan ke AWS CodeCommit repositori dengan kredensi berputar di Panduan Pengguna.AWS CodeCommit
- Akses Amazon Keyspaces (untuk Apache Cassandra) — Dalam situasi di mana Anda tidak dapat menggunakan pengguna di Pusat IAM Identitas, seperti untuk tujuan pengujian kompatibilitas

Cassandra, Anda dapat menggunakan IAM pengguna dengan kredensi khusus layanan untuk mengautentikasi dengan Amazon Keyspaces. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud Anda Akun AWS atau ke aplikasi cloud Anda. Anda juga dapat terhubung ke Amazon Keyspaces menggunakan kredensil sementara. Untuk informasi selengkapnya, lihat Menggunakan kredensi sementara untuk terhubung ke Amazon Keyspaces menggunakan IAM peran dan plugin SiGv4 di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

Ikuti praktik terbaik untuk melindungi kredensil pengguna root Anda

Saat Anda membuat Akun AWS, Anda membuat kredensi pengguna root untuk masuk ke file. AWS Management Console Lindungi kredenal pengguna root Anda dengan cara yang sama seperti Anda melindungi informasi pribadi sensitif lainnya. Untuk lebih memahami cara mengamankan dan menskalakan proses pengguna root Anda, lihatPraktik terbaik pengguna root untuk Anda Akun AWS.

Terapkan izin hak istimewa paling sedikit

Saat Anda menetapkan izin dengan IAM kebijakan, berikan hanya izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Anda dapat memulai dengan izin luas saat menjelajahi izin yang diperlukan untuk beban kerja atau kasus penggunaan Anda. Saat kasus penggunaan Anda matang, Anda dapat bekerja untuk mengurangi izin yang Anda berikan untuk bekerja menuju hak istimewa yang paling sedikit. Untuk informasi selengkapnya tentang penggunaan IAM untuk menerapkan izin, lihatKebijakan dan izin di AWS Identity and Access Management.

Memulai kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit

Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan oleh semua pelanggan. AWS Oleh karena itu, kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan terkelola pelanggan yang spesifik untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat AWS kebijakan terkelola. Untuk informasi selengkapnya

tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa paling sedikit berdasarkan aktivitas akses

Untuk hanya memberikan izin yang diperlukan untuk melakukan tugas, Anda dapat membuat kebijakan berdasarkan aktivitas akses yang masuk AWS CloudTrail. IAMAccess Analyzer menganalisis layanan dan tindakan yang digunakan IAM peran Anda, lalu menghasilkan kebijakan berbutir halus yang dapat Anda gunakan. Setelah menguji setiap kebijakan yang dihasilkan, Anda dapat menerapkan kebijakan tersebut ke lingkungan produksi. Ini memastikan bahwa Anda hanya memberikan izin yang diperlukan untuk beban kerja Anda. Untuk informasi selengkapnya tentang pembuatan kebijakan, lihat pembuatan kebijakan IAM Access Analyzer.

Tinjau dan hapus pengguna, peran, izin, kebijakan, dan kredensil yang tidak digunakan secara teratur

Anda mungkin memiliki IAM pengguna, peran, izin, kebijakan, atau kredensial yang tidak lagi Anda perlukan. Akun AWS IAMmenyediakan informasi yang terakhir diakses untuk membantu Anda mengidentifikasi pengguna, peran, izin, kebijakan, dan kredensi yang tidak lagi Anda perlukan sehingga Anda dapat menghapusnya. Ini membantu Anda mengurangi jumlah pengguna, peran, izin, kebijakan, dan kredensional yang harus Anda pantau. Anda juga dapat menggunakan informasi ini untuk menyempurnakan IAM kebijakan Anda agar lebih mematuhi izin hak istimewa paling sedikit. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Gunakan ketentuan dalam IAM kebijakan untuk membatasi akses lebih lanjut

Anda dapat menentukan kondisi di mana pernyataan kebijakan berlaku. Dengan begitu, Anda dapat memberikan akses ke tindakan dan sumber daya, tetapi hanya jika permintaan akses memenuhi persyaratan tertentu. Misalnya, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakanSSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan, tetapi hanya jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat IAMJSONelemen kebijakan: Condition.

Verifikasi akses publik dan lintas akun ke sumber daya dengan IAM Access Analyzer

Sebelum Anda memberikan izin untuk akses publik atau lintas akun AWS, kami sarankan Anda memverifikasi apakah akses tersebut diperlukan. Anda dapat menggunakan IAM Access Analyzer untuk membantu Anda melihat pratinjau dan menganalisis akses publik dan lintas akun untuk jenis sumber daya yang didukung. Anda melakukan ini dengan meninjau temuan yang dihasilkan IAM Access Analyzer. Temuan ini membantu Anda memverifikasi bahwa kontrol akses sumber daya Anda memberikan akses yang Anda harapkan. Selain itu, saat memperbarui izin publik dan lintas akun, Anda dapat memverifikasi efek perubahan sebelum menerapkan kontrol akses baru ke sumber daya Anda. IAMAccess Analyzer juga memantau jenis sumber daya yang didukung secara terus menerus dan menghasilkan temuan untuk sumber daya yang memungkinkan akses publik atau lintas akun. Untuk informasi selengkapnya, lihat Mempratinjau akses dengan IAM Access Analyzer APIs.

Gunakan IAM Access Analyzer untuk memvalidasi IAM kebijakan Anda guna memastikan izin yang aman dan fungsional

Validasi kebijakan yang Anda buat untuk memastikan bahwa IAMkebijakan tersebut mematuhi bahasa kebijakan (JSON) dan praktik IAM terbaik. Anda dapat memvalidasi kebijakan Anda dengan menggunakan validasi kebijakan IAM Access Analyzer. IAMAccess Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Saat Anda membuat kebijakan baru atau mengedit kebijakan yang ada di konsol, IAM Access Analyzer memberikan rekomendasi untuk membantu Anda menyempurnakan dan memvalidasi kebijakan sebelum menyimpannya. Selain itu, kami menyarankan Anda meninjau dan memvalidasi semua kebijakan yang ada. Untuk informasi selengkapnya, lihat Validasi kebijakan IAM Access Analyzer. Untuk informasi selengkapnya tentang pemeriksaan kebijakan yang disediakan oleh IAM Access Analyzer, lihat referensi pemeriksaan kebijakan IAM Access Analyzer.

Menetapkan pagar pembatas izin di beberapa akun

Saat Anda menskalakan beban kerja Anda, pisahkan dengan menggunakan beberapa akun yang dikelola. AWS Organizations Kami menyarankan Anda menggunakan kebijakan kontrol layanan Organizations (SCPs) untuk membuat pagar pembatas izin untuk mengontrol akses bagi semua IAM pengguna dan peran di seluruh akun Anda. SCPsadalah jenis kebijakan organisasi yang dapat Anda gunakan untuk mengelola izin di organisasi Anda di tingkat AWS organisasi, OU, atau akun. Pagar pembatas izin yang Anda buat berlaku untuk semua pengguna dan peran dalam akun yang

tercakup. Namun, SCPs saja tidak cukup untuk memberikan izin ke akun di organisasi Anda. Untuk melakukannya, administrator Anda harus melampirkan kebijakan berbasis identitas atau sumber daya ke IAM pengguna, IAM peran, atau sumber daya di akun Anda. Untuk informasi lebih lanjut, lihat AWS Organizations, akun, dan IAM pagar pembatas.

Gunakan batas izin untuk mendelegasikan manajemen izin dalam akun

Dalam beberapa skenario, Anda mungkin ingin mendelegasikan manajemen izin dalam akun kepada orang lain. Misalnya, Anda dapat mengizinkan pengembang untuk membuat dan mengelola peran untuk beban kerja mereka. Saat Anda mendelegasikan izin kepada orang lain, gunakan batas izin untuk menetapkan izin maksimum yang Anda delegasikan. Batas izin adalah fitur lanjutan untuk menggunakan kebijakan terkelola guna menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke peran. IAM Batas izin tidak memberikan izin sendiri. Untuk informasi selengkapnya, lihat Batas izin untuk entitas IAM.

Praktik terbaik pengguna root untuk Anda Akun AWS

Saat pertama kali membuat Akun AWS, Anda mulai dengan kumpulan kredensil default dengan akses lengkap ke semua AWS sumber daya di akun Anda. Identitas ini disebut pengguna Akun AWS root. Kami sangat menyarankan Anda untuk tidak mengakses pengguna Akun AWS root kecuali Anda memiliki tugas yang memerlukan kredensi pengguna root. Anda perlu mengamankan kredensi pengguna root dan mekanisme pemulihan akun Anda untuk membantu memastikan Anda tidak mengekspos kredenal Anda yang sangat istimewa untuk penggunaan yang tidak sah.

Alih-alih mengakses pengguna root, buat pengguna administratif untuk tugas sehari-hari.

- Jika Anda memiliki yang baru Akun AWS, lihatMenyiapkan Anda Akun AWS.
- Untuk beberapa yang Akun AWS dikelola AWS Organizations, lihat Menyiapkan Akun AWS akses untuk pengguna administratif Pusat IAM Identitas.

Dengan pengguna administratif Anda, Anda kemudian dapat membuat identitas tambahan untuk pengguna yang membutuhkan akses ke sumber daya di Anda Akun AWS. Kami sangat menyarankan Anda meminta pengguna untuk mengautentikasi dengan kredensi sementara saat mengakses. AWS

- Untuk satu, mandiri Akun AWS, gunakan IAMperan untuk membuat identitas di akun Anda dengan izin tertentu. Peran dimaksudkan untuk diasumsikan oleh siapa saja yang membutuhkannya. Juga, peran tidak memiliki kredensi jangka panjang standar, seperti kata sandi atau kunci

akses, yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, peran tersebut akan memberikan kredensial keamanan sementara untuk sesi peran. Tidak seperti IAM peran, IAMpengguna memiliki kredensi jangka panjang seperti kata sandi dan kunci akses. Jika memungkinkan, praktik terbaik merekomendasikan untuk mengandalkan kredensil sementara alih-alih membuat IAM pengguna yang memiliki kredensi jangka panjang seperti kata sandi dan kunci akses.

- Untuk beberapa yang Akun AWS dikelola melalui Organizations, gunakan pengguna tenaga kerja Pusat IAM Identitas. Dengan Pusat IAM Identitas, Anda dapat mengelola pengguna secara terpusat di seluruh Anda Akun AWS dan izin dalam akun tersebut. Kelola identitas pengguna Anda dengan Pusat IAM Identitas atau dari penyedia identitas eksternal. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center dalam Panduan Pengguna AWS IAM Identity Center .

Topik

Amankan kredensi pengguna root Anda untuk mencegah penggunaan yang tidak sah

Amankan kredensi pengguna root Anda dan gunakan hanya untuk tugas-tugas yang membutuhkannya. Untuk membantu mencegah penggunaan yang tidak sah, jangan bagikan kata sandi pengguna root AndaMFA, kunci akses, pasangan CloudFront kunci, atau menandatangani sertifikat dengan siapa pun, kecuali mereka yang memiliki kebutuhan bisnis yang ketat untuk mengakses pengguna root.

Jangan menyimpan kata sandi pengguna root dengan alat yang bergantung Layanan AWS pada akun yang diakses menggunakan kata sandi yang sama. Jika Anda kehilangan atau lupa kata

sandi pengguna root Anda, Anda tidak akan dapat mengakses alat ini. Kami menyarankan Anda memprioritaskan ketahanan dan mempertimbangkan untuk mewajibkan dua orang atau lebih untuk mengotorisasi akses ke lokasi penyimpanan. Akses ke kata sandi atau lokasi penyimpanannya harus dicatat dan dipantau.

Gunakan kata sandi pengguna root yang kuat untuk membantu melindungi akses

Kami menyarankan Anda menggunakan kata sandi yang kuat dan unik. Alat seperti pengelola kata sandi dengan algoritme pembuatan kata sandi yang kuat dapat membantu Anda mencapai tujuan ini. AWS mengharuskan kata sandi Anda memenuhi ketentuan berikut:

- Itu harus memiliki minimal 8 karakter dan maksimal 128 karakter.
- Ini harus mencakup minimal tiga dari campuran tipe karakter berikut: huruf besar, huruf kecil, angka, dan! @ # $% ^ & * () <> [] {} | _+-= simbol.
- Itu tidak boleh identik dengan Akun AWS nama atau alamat email Anda.

Untuk informasi selengkapnya, lihat Ubah kata sandi untuk Pengguna root akun AWS.

Amankan login pengguna root Anda dengan otentikasi multi-faktor () MFA

Karena pengguna root dapat melakukan tindakan istimewa, penting MFA untuk menambahkan pengguna root sebagai faktor otentikasi kedua selain alamat email dan kata sandi sebagai kredensi masuk. Kami sangat menyarankan MFA untuk mengaktifkan beberapa kredensil pengguna root Anda untuk memberikan fleksibilitas dan ketahanan tambahan dalam strategi keamanan Anda. Anda dapat mendaftarkan hingga delapan MFA perangkat dari kombinasi MFA jenis yang saat ini didukung dengan pengguna Akun AWS root Anda.

- FIDOKunci keamanan perangkat keras bersertifikat disediakan oleh penyedia pihak ketiga. Untuk informasi selengkapnya, lihat Mengaktifkan kunci FIDO keamanan untuk pengguna Akun AWS root.
- Perangkat keras yang menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali () berbasis waktu. TOTP Untuk informasi selengkapnya, lihat Mengaktifkan TOTP token perangkat keras untuk pengguna Akun AWS root.
- Aplikasi otentikator virtual yang berjalan di ponsel atau perangkat lain dan mengemulasi perangkat fisik. Untuk informasi selengkapnya, lihat Mengaktifkan MFA perangkat virtual untuk pengguna Akun AWS root Anda.

Jangan membuat kunci akses untuk pengguna root

Kunci akses memungkinkan Anda menjalankan AWS perintah di Command Line Interface (AWS CLI) atau menggunakan API operasi dari salah satu AWS SDKs. Kami sangat menyarankan agar Anda tidak membuat pasangan kunci akses untuk pengguna root Anda karena pengguna root memiliki akses penuh ke semua Layanan AWS dan sumber daya di akun, termasuk informasi penagihan.

Karena hanya beberapa tugas yang memerlukan pengguna root dan Anda biasanya jarang melakukan tugas-tugas tersebut, kami sarankan masuk ke AWS Management Console untuk melakukan tugas pengguna root. Sebelum membuat kunci akses, tinjau alternatif untuk kunci akses jangka panjang.

Gunakan persetujuan multi-orang untuk login pengguna root sedapat mungkin

Pertimbangkan untuk menggunakan persetujuan multi-orang untuk memastikan bahwa tidak ada satu orang pun yang dapat mengakses keduanya MFA dan kata sandi untuk pengguna root. Beberapa perusahaan menambahkan lapisan keamanan tambahan dengan menyiapkan satu grup administrator dengan akses ke kata sandi, dan grup administrator lain dengan akses ke. MFA Satu anggota dari setiap grup harus berkumpul untuk masuk sebagai pengguna root.

Menggunakan alamat email grup untuk kredensi pengguna root

Gunakan alamat email yang dikelola oleh bisnis Anda dan teruskan pesan yang diterima langsung ke sekelompok pengguna. Jika AWS harus menghubungi pemilik akun, pendekatan ini mengurangi risiko keterlambatan dalam menanggapi, bahkan jika individu sedang berlibur, sakit, atau telah meninggalkan bisnis. Alamat email yang digunakan untuk pengguna root tidak boleh digunakan untuk tujuan lain.

Batasi akses ke mekanisme pemulihan akun

Pastikan Anda mengembangkan proses untuk mengelola mekanisme pemulihan kredensi pengguna root jika Anda memerlukan akses ke sana selama keadaan darurat seperti pengambilalihan akun administratif Anda.

- Pastikan Anda memiliki akses ke kotak masuk email pengguna root Anda sehingga Anda dapat mengatur ulang kata sandi pengguna root yang hilang atau terlupakan.
- Jika MFA untuk pengguna Akun AWS root Anda hilang, rusak, atau tidak berfungsi, Anda dapat masuk menggunakan yang lain yang MFA terdaftar ke kredensi pengguna root yang sama.

Jika Anda kehilangan akses ke semuaMFAs, Anda memerlukan nomor telepon dan email yang digunakan untuk mendaftarkan akun Anda, agar up to date dan dapat diakses untuk memulihkan akun AndaMFA. Untuk detailnya, lihat Memulihkan MFA perangkat pengguna root.

- Jika Anda memilih untuk tidak menyimpan kata sandi pengguna root Anda danMFA, maka nomor telepon yang terdaftar di akun Anda dapat digunakan sebagai cara alternatif untuk memulihkan kredensi pengguna root. Pastikan Anda memiliki akses ke nomor telepon kontak, perbarui nomor telepon, dan batasi siapa yang memiliki akses untuk mengelola nomor telepon.

Tidak seorang pun harus memiliki akses ke kotak masuk email dan nomor telepon karena keduanya adalah saluran verifikasi untuk memulihkan kata sandi pengguna root Anda. Penting untuk memiliki dua kelompok individu yang mengelola saluran ini. Satu grup memiliki akses ke alamat email utama Anda dan grup lain yang memiliki akses ke nomor telepon utama untuk memulihkan akses ke akun Anda sebagai pengguna root.

Amankan kredensi pengguna root akun Organizations Anda

Saat Anda beralih ke strategi multi-akun dengan Organizations, masing-masing Akun AWS memiliki kredensi pengguna root sendiri yang perlu Anda amankan. Akun yang Anda gunakan untuk membuat organisasi Anda adalah akun manajemen dan akun lainnya di organisasi Anda adalah akun anggota.

Amankan kredensi pengguna root untuk akun anggota

Jika Anda menggunakan Organizations untuk mengelola beberapa akun, ada dua strategi yang dapat Anda ambil untuk mengamankan akses pengguna root di Organizations Anda.

- Amankan kredensi pengguna root dari akun Organizations Anda dengan. MFA
- Jangan mengatur ulang kata sandi pengguna root untuk akun Anda, dan hanya memulihkan akses ke sana bila diperlukan menggunakan proses reset kata sandi. Saat Anda membuat akun anggota di organisasi Anda, Organizations secara otomatis membuat IAM peran dalam akun anggota yang memungkinkan akun manajemen akses sementara ke akun anggota.

Untuk detailnya, lihat Mengakses akun anggota di organisasi Anda di Panduan Pengguna Organizations.

Menetapkan kontrol keamanan preventif di Organizations menggunakan kebijakan kontrol layanan () SCP

Jika Anda menggunakan Organizations untuk mengelola beberapa akun, Anda dapat menerapkan SCP untuk membatasi akses ke pengguna root akun anggota. Menyangkal semua tindakan pengguna root di akun anggota Anda, kecuali untuk tindakan khusus root tertentu, membantu mencegah akses yang tidak sah. Untuk detailnya, lihat Menggunakan SCP untuk membatasi apa yang dapat dilakukan pengguna root di akun anggota Anda.

Memantau akses dan penggunaan

Kami menyarankan Anda menggunakan mekanisme pelacakan saat ini untuk memantau, memperingatkan, dan melaporkan login dan penggunaan kredenal pengguna root, termasuk peringatan yang mengumumkan login dan penggunaan pengguna root. Layanan berikut dapat membantu memastikan bahwa penggunaan kredensi pengguna root dilacak dan melakukan pemeriksaan keamanan yang dapat membantu mencegah penggunaan yang tidak sah.

- Jika Anda ingin diberi tahu tentang aktivitas login pengguna root di akun Anda, Anda dapat memanfaatkan Amazon CloudWatch untuk membuat aturan Acara yang mendeteksi kapan kredensil pengguna root digunakan dan memicu pemberitahuan ke administrator keamanan Anda. Untuk detailnya, lihat Memantau dan memberi tahu aktivitas pengguna Akun AWS root.
- Jika Anda ingin mengatur notifikasi untuk mengingatkan Anda tentang tindakan pengguna root yang disetujui, Anda dapat memanfaatkan Amazon EventBridge bersama dengan Amazon SNS untuk menulis EventBridge aturan untuk melacak penggunaan pengguna root untuk tindakan tertentu dan memberi tahu Anda menggunakan SNS topik Amazon. Sebagai contoh, lihat Mengirim pemberitahuan saat objek Amazon S3 dibuat.
- Jika Anda sudah menggunakan GuardDuty sebagai layanan deteksi ancaman, Anda dapat memperluas kemampuannya untuk memberi tahu Anda ketika kredensi pengguna root digunakan di akun Anda.

Peringatan harus mencakup, tetapi tidak terbatas pada, alamat email untuk pengguna root. Memiliki prosedur untuk bagaimana menanggapi peringatan sehingga personel yang menerima peringatan akses pengguna root memahami cara memvalidasi bahwa akses pengguna root diharapkan, dan bagaimana meningkatkan jika mereka yakin bahwa insiden keamanan sedang berlangsung. Untuk contoh cara mengonfigurasi peringatan, lihat Memantau dan memberi tahu aktivitas pengguna Akun AWS root.

Evaluasi MFA kepatuhan pengguna root

- AWS Config menggunakan aturan untuk membantu menegakkan praktik terbaik pengguna root. Anda dapat menggunakan aturan AWS terkelola untuk mengharuskan pengguna root mengaktifkan autentikasi multi-faktor (MFA). AWS Config juga dapat mengidentifikasi kunci akses untuk pengguna root.
- Security Hub memberi Anda pandangan komprehensif tentang status keamanan Anda AWS dan membantu Anda menilai AWS lingkungan Anda terhadap standar industri keamanan dan praktik terbaik, seperti memiliki MFA pada pengguna root dan tidak memiliki kunci akses pengguna root. Untuk detail tentang aturan yang tersedia, lihat AWS Identity and Access Management kontrol di Panduan Pengguna Security Hub.
- Trusted Advisor menyediakan pemeriksaan keamanan sehingga Anda tahu apakah MFA tidak diaktifkan pada akun pengguna root. Untuk informasi selengkapnya, lihat MFAdi Root Account di AWS Support User Guide.

Jika Anda perlu melaporkan masalah keamanan di akun Anda, lihat Melaporkan Email Mencurigakan atau Pelaporan Kerentanan. Atau, Anda dapat menghubungi AWS untuk bantuan dan panduan tambahan.

Kasus penggunaan bisnis untuk IAM

Kasus penggunaan bisnis sederhana untuk IAM dapat membantu Anda memahami cara-cara dasar Anda dapat menerapkan layanan untuk mengontrol AWS akses yang dimiliki pengguna Anda. Kasus penggunaan dijelaskan secara umum, tanpa mekanisme bagaimana Anda akan menggunakan IAM API untuk mencapai hasil yang Anda inginkan.

Kasus penggunaan ini melihat dua cara khas yang mungkin digunakan oleh perusahaan fiksi bernama Example Corp. IAM Skenario pertama mempertimbangkan Amazon Elastic Compute Cloud (AmazonEC2). Kedua, Amazon Simple Storage Service (Amazon S3).

Untuk informasi selengkapnya tentang penggunaan IAM dengan layanan lain dari AWS, lihatAWS layanan yang bekerja dengan IAM.

Topik

- Pengaturan awal example corp
- Kasus penggunaan untuk IAM Amazon EC2

- Kasus penggunaan untuk IAM Amazon S3

Pengaturan awal example corp

Nikki Wolf dan Mateo Jackson adalah pendiri Example Corp. Setelah memulai perusahaan, mereka membuat Akun AWS dan mengatur AWS IAM Identity Center(Pusat IAM Identitas) untuk membuat akun administratif untuk digunakan dengan sumber daya mereka. AWS Saat Anda mengatur akses akun untuk pengguna administratif, Pusat IAM Identitas akan membuat IAM peran yang sesuai. Peran ini, yang dikendalikan oleh Pusat IAM Identitas, dibuat dalam peran yang relevan Akun AWS, dan kebijakan yang ditentukan dalam kumpulan AdministratorAccessizin dilampirkan ke peran.

Karena mereka sekarang memiliki akun administrator, Nikki dan Mateo tidak perlu lagi menggunakan pengguna root mereka untuk mengakses akun mereka. Akun AWS Mereka berencana untuk hanya menggunakan pengguna root untuk menyelesaikan tugas-tugas yang hanya dapat dilakukan oleh pengguna root. Setelah meninjau praktik terbaik keamanan, mereka mengonfigurasi otentikasi multi-faktor (MFA) untuk kredensil pengguna root mereka dan memutuskan cara melindungi kredenal pengguna root mereka.

Seiring pertumbuhan perusahaan mereka, mereka mempekerjakan karyawan untuk bekerja sebagai pengembang, admin, penguji, manajer, dan administrator sistem. Nikki bertanggung jawab atas operasi, sementara Mateo mengelola tim teknik. Mereka menyiapkan Server Domain Direktori Aktif untuk mengelola akun karyawan dan mengelola akses ke sumber daya internal perusahaan.

Untuk memberikan karyawan mereka akses ke AWS sumber daya, mereka menggunakan IAM Identity Center untuk menghubungkan Active Directory perusahaan mereka ke mereka Akun AWS.

Karena mereka menghubungkan Active Directory ke IAM Identity Center, pengguna, grup, dan keanggotaan grup disinkronkan dan ditentukan. Mereka harus menetapkan set izin dan peran ke grup yang berbeda untuk memberikan pengguna tingkat akses yang benar ke AWS sumber daya. Mereka menggunakan AWS kebijakan terkelola untuk fungsi pekerjaan dalam AWS Management Console untuk membuat set izin ini:

- Administrator
- Penagihan
- Pengembang
- Administrator jaringan
- Administrator basis data

- Administrator sistem
- Support pengguna

Kemudian mereka menetapkan set izin ini ke peran yang ditetapkan ke grup Direktori Aktif mereka.

Untuk step-by-step panduan yang menjelaskan konfigurasi awal Pusat IAM Identitas, lihat Memulai di Panduan AWS IAM Identity Center Pengguna. Untuk informasi selengkapnya tentang penyediaan akses pengguna Pusat IAM Identitas, lihat Akses masuk tunggal ke AWS akun di Panduan Pengguna.AWS IAM Identity Center

Kasus penggunaan untuk IAM Amazon EC2

Perusahaan seperti Example Corp biasanya menggunakan IAM untuk berinteraksi dengan layanan seperti AmazonEC2. Untuk memahami bagian dari kasus penggunaan ini, Anda memerlukan pemahaman dasar tentang AmazonEC2. Untuk informasi lebih lanjut tentang AmazonEC2, buka Panduan EC2 Pengguna Amazon.

EC2Izin Amazon untuk grup pengguna

Untuk memberikan kontrol “perimeter”, Nikki melampirkan kebijakan ke grup AllUsers pengguna. Kebijakan ini menolak AWS permintaan apa pun dari pengguna jika alamat IP asal berada di luar jaringan perusahaan Example Corp.

Di Example Corp, IAM grup yang berbeda memerlukan izin yang berbeda:

- Administrator sistem — Perlu izin untuk membuat dan mengelolaAMIs, instance, snapshot, volume, grup keamanan, dan sebagainya. Nikki melampirkan kebijakan `AmazonEC2FullAccess` AWS terkelola ke grup SysAdmins pengguna yang memberi anggota grup izin untuk menggunakan semua tindakan AmazonEC2.
- Pengembang – Perlu kemampuan untuk bekerja hanya dengan instans. Oleh karena itu Mateo membuat dan melampirkan kebijakan ke grup pengguna Pengembang yang memungkinkan pengembang untuk meneleponDescribeInstances,,RunInstances, StopInstancesStartInstances, dan. TerminateInstances

> **Note**
>
> Amazon EC2 menggunakan SSH kunci, kata sandi Windows, dan grup keamanan untuk mengontrol siapa yang memiliki akses ke sistem operasi EC2 instans Amazon tertentu.

Tidak ada metode dalam IAM sistem untuk mengizinkan atau menolak akses ke sistem operasi dari instance tertentu.

- Mendukung pengguna - Seharusnya tidak dapat melakukan EC2 tindakan Amazon apa pun kecuali mencantumkan EC2 sumber daya Amazon yang saat ini tersedia. Oleh karena itu, Nikki membuat dan melampirkan kebijakan ke grup pengguna Support yang hanya memungkinkan mereka memanggil operasi EC2 “Deskripsikan” API Amazon.

Untuk contoh seperti apa kebijakan ini, lihat Contoh kebijakan berbasis identitas IAM dan Menggunakan AWS Identity and Access Management di Panduan EC2 Pengguna Amazon.

Perubahan fungsi pekerjaan pengguna

Pada titik tertentu, salah satu pengembang, Paulo Santos, mengubah fungsi pekerjaan dan menjadi manajer. Sebagai manajer, Paulo menjadi bagian dari kelompok pengguna Support sehingga ia dapat membuka kasus dukungan untuk pengembangnya. Mateo memindahkan Paulo dari grup pengguna Pengembang ke grup pengguna Support. Sebagai hasil dari langkah ini, kemampuannya untuk berinteraksi dengan EC2 instans Amazon terbatas. Dia tidak dapat meluncurkan atau memulai instans. Dia juga tidak dapat menghentikan atau mematikan instans yang ada, bahkan jika dia pengguna yang meluncurkan atau memulai instans. Dia hanya dapat mencantumkan instans yang telah diluncurkan pengguna Example Corp.

Kasus penggunaan untuk IAM Amazon S3

Perusahaan seperti Example Corp juga biasanya akan menggunakan IAM Amazon S3. John telah menciptakan ember Amazon S3 untuk perusahaan bernama amzn-s3-demo-bucket.

Pembuatan pengguna dan grup pengguna lain

Sebagai karyawan, Zhang Wei dan Mary Major masing-masing harus dapat membuat data mereka sendiri di ember perusahaan. Mereka juga perlu membaca dan menulis data yang dikerjakan oleh semua developer. Untuk mengaktifkan ini, Mateo secara logis mengatur data dalam amzn-s3-demo-bucket menggunakan skema key prefix Amazon S3 seperti yang ditunjukkan pada gambar berikut.

```
/amzn-s3-demo-bucket
    /home
        /zhang
        /major
    /share
```

```
        /developers
        /managers
```

Mateo membagi `/amzn-s3-demo-bucket` menjadi satu set direktori rumah untuk setiap karyawan, dan area bersama untuk kelompok pengembang dan manajer.

Sekarang Mateo membuat serangkaian kebijakan untuk menetapkan izin kepada pengguna dan grup pengguna:

- Akses direktori rumah untuk Zhang — Mateo melampirkan kebijakan ke Wei yang memungkinkannya membaca, menulis, dan membuat daftar objek apa pun dengan awalan key Amazon S3 `/amzn-s3-demo-bucket/home/zhang/`
- Akses direktori rumah untuk Mayor - Mateo melampirkan kebijakan ke Mary yang memungkinkannya membaca, menulis, dan membuat daftar objek apa pun dengan awalan key Amazon S3 `/amzn-s3-demo-bucket/home/major/`
- Akses direktori bersama untuk grup pengguna pengembang — Mateo melampirkan kebijakan ke grup pengguna yang memungkinkan pengembang membaca, menulis, dan mencantumkan objek apa pun di `/amzn-s3-demo-bucket/share/developers/`
- Akses direktori bersama untuk grup pengguna manajer — Mateo melampirkan kebijakan ke grup pengguna yang memungkinkan manajer membaca, menulis, dan mencantumkan objek di `/amzn-s3-demo-bucket/share/managers/`

> **Note**
>
> Amazon S3 tidak secara otomatis memberi pengguna yang membuat izin bucket atau objek untuk melakukan tindakan lain pada bucket atau objek tersebut. Oleh karena itu, dalam IAM kebijakan Anda, Anda harus secara eksplisit memberikan izin kepada pengguna untuk menggunakan sumber daya Amazon S3 yang mereka buat.

Untuk contoh seperti apa kebijakan ini, lihat Kontrol Akses di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon. Untuk informasi tentang bagaimana kebijakan dievaluasi pada waktu aktif, lihat Logika evaluasi kebijakan.

Perubahan fungsi pekerjaan pengguna

Pada titik tertentu, salah satu pengembang, Zhang Wei, mengubah fungsi pekerjaan dan menjadi manajer. Kami berasumsi bahwa dia tidak lagi memerlukan akses ke dokumen dalam direktori

`share/developers`. Mateo, sebagai admin, memindahkan Wei ke grup `Managers` pengguna dan keluar dari grup `Developers` pengguna. Hanya dengan penugasan ulang sederhana itu, Wei secara otomatis mendapatkan semua izin yang diberikan kepada grup `Managers` pengguna, tetapi tidak dapat lagi mengakses data di direktori. `share/developers`

Integrasi dengan bisnis pihak ketiga

Organisasi sering kali bekerja dengan perusahaan mitra, konsultan, dan kontraktor. Contoh Corp memiliki mitra yang disebut Perusahaan Widget, dan karyawan Perusahaan Widget bernama Shirley Rodriguez perlu memasukkan data ke dalam ember untuk penggunaan Example Corp. Nikki membuat grup pengguna yang disebut WidgetCodan pengguna bernama `Shirley` dan menambahkan Shirley ke grup pengguna. WidgetCo Nikki juga membuat ember khusus yang disebut amzn-s3-demo-bucket1 untuk digunakan Shirley.

Nikki memperbarui kebijakan yang ada atau menambahkan yang baru untuk mengakomodasi mitra Widget Perusahaan. Misalnya, Nikki dapat membuat kebijakan baru yang menyangkal kemampuan anggota grup WidgetCo pengguna untuk menggunakan tindakan apa pun selain menulis. Kebijakan ini hanya akan diperlukan jika terdapat kebijakan luas yang memberi semua pengguna akses ke serangkaian tindakan Amazon S3.

IAMtutorial

Tutorial berikut menyajikan end-to-end prosedur lengkap untuk tugas-tugas umum untuk AWS Identity and Access Management (IAM). Tutorial ini dimaksudkan untuk lingkungan tipe lab, dengan nama perusahaan fiktif , nama pengguna fiktif, dan sebagainya. Tujuannya adalah untuk memberikan pedoman umum. Produk ini tidak dimaksudkan untuk penggunaan langsung di lingkungan produksi tanpa tinjauan dan adaptasi cermat untuk memenuhi kebutuhan unik di lingkungan organisasi Anda.

Tutorial

- IAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran
- IAMtutorial: Buat dan lampirkan kebijakan terkelola pelanggan pertama Anda
- IAMtutorial: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag
- IAMtutorial: Izinkan pengguna untuk mengelola kredensi dan pengaturan mereka MFA

IAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran

> **Important**
>
> IAMpraktik terbaik merekomendasikan bahwa Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensil sementara alih-alih menggunakan IAM pengguna dengan kredensil jangka panjang. Kami menyarankan Anda hanya menggunakan IAM pengguna untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.

Tutorial ini mengajarkan Anda cara menggunakan peran untuk mendelegasikan akses ke sumber daya yang berbeda Akun AWS disebut Destinasi dan Originasi. Anda berbagi sumber daya di satu akun dengan pengguna di akun yang berbeda. Dengan mengatur akses lintas akun dengan cara ini, Anda tidak perlu membuat IAM pengguna individu di setiap akun. Selain itu, pengguna tidak perlu keluar dari satu akun dan masuk ke akun lain untuk mengakses sumber daya di akun yang berbeda Akun AWS. Setelah mengonfigurasi peran, Anda melihat cara menggunakan peran dari AWS Management Console, AWS CLI, danAPI.

Dalam tutorial ini, akun Tujuan mengelola data aplikasi yang diakses oleh berbagai aplikasi dan tim. Di setiap akun, Anda menyimpan informasi aplikasi di ember Amazon S3. Anda mengelola IAM pengguna di akun Originating, di mana Anda memiliki dua peran IAM pengguna: Pengembang dan Analis. Pengembang dan Analis menggunakan akun Originating untuk menghasilkan data yang dibagikan oleh beberapa layanan mikro. Kedua peran memiliki izin untuk bekerja di akun Originating dan mengakses sumber daya di sana. Dari waktu ke waktu, pengembang harus memperbarui data bersama di akun Tujuan. Pengembang menyimpan data ini dalam ember Amazon S3 yang disebut. `amzn-s3-demo-bucket-shared-container`

Di akhir tutorial ini, Anda memiliki hal berikut:

- Pengguna di akun Originating (akun tepercaya) diizinkan untuk mengambil peran tertentu dalam akun Tujuan.
- Peran di akun Tujuan (akun kepercayaan) diizinkan untuk mengakses bucket Amazon S3 tertentu.
- `amzn-s3-demo-bucket-shared-container`Bucket di akun Tujuan.

Pengembang dapat menggunakan peran dalam AWS Management Console untuk mengakses `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. Mereka juga dapat mengakses bucket dengan menggunakan API panggilan yang diautentikasi oleh kredensi sementara yang disediakan oleh peran. Upaya serupa oleh seorang Analis untuk menggunakan peran gagal.

Alur kerja ini memiliki tiga langkah dasar:

Membuat peran di Akun Tujuan

Pertama, Anda menggunakan AWS Management Console untuk membangun kepercayaan antara akun Tujuan (nomor ID 999999999999) dan akun Originating (nomor ID 1111111111). Anda mulai dengan membuat IAM peran bernama UpdateData. Saat membuat peran, Anda menentukan akun Originating sebagai entitas tepercaya dan menentukan kebijakan izin yang memungkinkan pengguna tepercaya memperbarui bucket. `amzn-s3-demo-bucket-shared-container`

Berikan akses ke peran tersebut

Di bagian ini, Anda mengubah kebijakan peran untuk menolak akses Analis ke `UpdateData` peran tersebut. Karena Analis memiliki PowerUser akses dalam skenario ini, dan Anda harus secara eksplisit menolak kemampuan untuk menggunakan peran tersebut.

Akses uji dengan mengalihkan peran

Terakhir, sebagai Pengembang, Anda menggunakan `UpdateData` peran untuk memperbarui `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. Anda melihat cara mengakses peran melalui AWS konsol, AWS CLI, danAPI.

Pertimbangan

Sebelum Anda menggunakan IAM peran untuk mendelegasikan akses sumber daya Akun AWS, penting untuk mempertimbangkan hal berikut:

- Anda tidak dapat beralih ke peran ketika Anda masuk sebagai Pengguna root akun AWS.
- IAMperan dan kebijakan berbasis sumber daya mendelegasikan akses di seluruh akun hanya dalam satu partisi. Misalnya, anggap Anda memiliki akun di AS Barat (N. California) dalam partisi `aws` standar. Anda juga memiliki akun di Tiongkok (Beijing) dalam partisi `aws-cn`. Anda tidak dapat menggunakan kebijakan berbasis sumber daya Amazon S3 di akun Anda di Tiongkok (Beijing) untuk memungkinkan akses bagi pengguna dalam akun `aws` standar Anda.
- Anda dapat menggunakan AWS IAM Identity Center untuk memfasilitasi single sign-on (SSO) untuk eksternal Akun AWS (akun di luar AWS Organizations) menggunakan Security Assertion Markup Language (). SAML Untuk detailnya, lihat Mengintegrasikan eksternal Akun AWS ke AWS IAM Identity Center untuk manajemen akses pusat dengan penagihan independen menggunakan 2.0 SAML
- Anda dapat mengaitkan peran dengan AWS sumber daya seperti EC2 contoh Amazon atau AWS Lambda fungsi. Untuk detailnya, lihat Membuat peran untuk mendelegasikan izin ke layanan AWS.
- Jika Anda ingin memiliki aplikasi mengambil peran dalam aplikasi lain Akun AWS, Anda dapat menggunakan AWS SDKuntuk asumsi peran lintas akun. Untuk informasi selengkapnya, lihat Otentikasi dan akses di AWS SDKsdan Panduan Referensi Alat.
- Beralih peran menggunakan AWS Management Console hanya berfungsi dengan akun yang tidak memerlukan`ExternalId`. Misalnya, Anda memberikan akses untuk akun Anda ke pihak ketiga dan memerlukan `ExternalId` dalam elemen `Condition` dalam kebijakan izin Anda. Dalam hal ini, pihak ketiga dapat mengakses akun Anda hanya dengan menggunakan AWS APIatau alat baris perintah. Pihak ketiga tidak dapat menggunakan konsol karena harus memberikan nilai untuk`ExternalId`. Untuk informasi selengkapnya tentang skenario iniAkses ke Akun AWS yang dimiliki oleh pihak ketiga, lihat, dan Cara mengaktifkan akses lintas akun ke AWS Management Consoledi AWS Blog Keamanan.

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah melakukan hal berikut:

- Dua terpisah Akun AWS yang dapat Anda gunakan, satu untuk mewakili akun Originating, dan satu untuk mewakili akun Tujuan.
- Pengguna dan peran dalam akun Originating dibuat dan dikonfigurasi sebagai berikut:

Jabatan/Job	Pengguna	Izin
Developer	David	Kedua pengguna dapat masuk dan menggunakan AWS Management Console di akun Originating.
Analis	Jane	

- Anda tidak perlu membuat pengguna apa pun di akun Tujuan.
- Bucket Amazon S3 yang dibuat di akun Tujuan. Anda dapat menyebutnya `amzn-s3-demo-bucket-shared-container` dalam tutorial ini, tetapi karena nama bucket S3 harus unik secara global, Anda harus menggunakan bucket dengan nama yang berbeda.

Membuat peran di Akun Tujuan

Anda dapat mengizinkan pengguna dari satu Akun AWS untuk mengakses sumber daya di tempat lain Akun AWS. Dalam tutorial ini, kita akan melakukan ini dengan membuat peran yang mendefinisikan siapa yang dapat mengaksesnya dan izin apa yang diberikannya kepada pengguna yang beralih ke sana.

Pada langkah tutorial ini, Anda membuat peran di akun Tujuan dan menentukan akun Originating sebagai entitas tepercaya. Anda juga membatasi izin peran untuk hanya membaca dan menulis akses ke `amzn-s3-demo-bucket-shared-container` bucket. Siapa pun yang diberi izin untuk menggunakan peran dapat membaca dan menulis ke `shared-container` ember.

Sebelum Anda dapat membuat peran, Anda memerlukan ID akun dari Originating Akun AWS. Masing-masing Akun AWS memiliki pengenal ID akun unik yang ditetapkan untuk itu.

Untuk mendapatkan Originating Akun AWS ID

1. Masuk ke AWS Management Console sebagai administrator akun Originating, dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di IAM konsol, pilih nama pengguna Anda di bilah navigasi di kanan atas. Itu terlihat seperti ini: ***username***@***account_ID_number_or_alias***.

   Untuk skenario ini, Anda dapat menggunakan ID akun 111111111111 untuk akun Originating. Namun, Anda harus menggunakan ID akun yang valid jika Anda menggunakan skenario ini di lingkungan pengujian Anda.

Untuk membuat peran di akun Tujuan yang dapat digunakan oleh akun Originating

1. Masuk ke AWS Management Console sebagai administrator akun Tujuan, dan buka IAM konsol.
2. Sebelum membuat peran, siapkan kebijakan terkelola yang menentukan izin untuk persyaratan peran. Anda kemudian perlu melampirkan kebijakan ini ke peran tersebut pada langkah berikutnya.

   Anda ingin mengatur akses baca dan tulis ke bucket `amzn-s3-demo-bucket-shared-container`. Meskipun AWS menyediakan beberapa kebijakan terkelola Amazon S3, tidak ada yang menyediakan akses baca dan tulis ke satu bucket Amazon S3. Anda dapat membuat kebijakan Anda sendiri.

   Pada panel navigasi, silakan pilih Kebijakan dan kemudian pilih Buat kebijakan.
3. Pilih JSONtab dan salin teks dari dokumen JSON kebijakan berikut. Rekatkan teks ini ke dalam kotak JSONteks, ganti resource ARN (`arn:aws:s3:::shared-container`) dengan yang asli untuk bucket Amazon S3 Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container"
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container/*"
    }
  ]
}
```

`ListAllMyBuckets`Tindakan memberikan izin untuk mencantumkan semua bucket yang dimiliki oleh pengirim permintaan yang diautentikasi. Izin `ListBucket` memungkinkan pengguna melihat objek dalam bucket `amzn-s3-demo-bucket-shared-container`. Izin `GetObject`, `PutObject`, `DeleteObject` memungkinkan pengguna melihat, memperbarui, dan menghapus konten di bucket `amzn-s3-demo-bucket-shared-container`.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

4. Pada halaman Tinjau dan buat, ketik **`read-write-app-bucket`** nama kebijakan. Tinjau izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul di daftar kebijakan terkelola.

5. Pada panel navigasi, silakan pilih Peran lalu pilih Buat peran.
6. Pilih An Akun AWStipe peran.
7. Untuk Account ID, ketik Originating Account ID.

   Tutorial ini menggunakan contoh ID akun **111111111111** untuk akun Originating. Anda harus menggunakan ID akun yang valid. Jika Anda menggunakan ID akun yang tidak valid, seperti**111111111111**, IAM tidak memungkinkan Anda membuat peran baru.

Untuk saat ini Anda tidak perlu memerlukan ID eksternal, atau mengharuskan pengguna untuk memiliki otentikasi multi-faktor (MFA) untuk mengambil peran. Biarkan opsi ini tidak dipilih. Untuk informasi selengkapnya, lihat AWS Otentikasi multi-faktor di IAM.

8. Pilih Berikutnya: Izin untuk mengatur izin yang terkait dengan peran.
9. Pilih kotak centang di samping kebijakan yang Anda buat sebelumnya.

> **Kiat**
>
> Untuk Filter, pilih Pelanggan berhasil memfilter daftar agar hanya menyertakan kebijakan yang Anda buat. Ini menyembunyikan AWS membuat kebijakan dan membuatnya lebih mudah untuk menemukan yang Anda butuhkan.

Lalu, pilih Selanjutnya.

10. (Opsional) Tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
11. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
12. Setelah meninjau peran, klik Buat peran.

Peran `UpdateData` muncul di daftar peran.

Sekarang Anda harus mendapatkan Amazon Resource Name (ARN) peran, pengidentifikasi unik untuk peran tersebut. Saat mengubah peran Pengembang di akun Originating, Anda menentukan peran ARN dari akun Tujuan untuk memberikan atau menolak izin.

Untuk mendapatkan ARN untuk UpdateData

1. Di panel navigasi IAM konsol, pilih Peran.
2. Dalam daftar peran, pilih peran `UpdateData`.
3. Di bagian Ringkasan panel detail, salin ARN nilai Peran.

Akun Tujuan memiliki ID akun 999999999999, jadi perannya adalah. ARN `arn:aws:iam::999999999999:role/UpdateData` Pastikan Anda memberikan yang nyata Akun AWS ID untuk akun Tujuan.

Pada titik ini, Anda telah membangun kepercayaan antara akun Tujuan dan Asal. Anda melakukan ini dengan membuat peran di akun Tujuan yang mengidentifikasi akun Originating sebagai prinsipal tepercaya. Anda juga menentukan apa yang dapat dilakukan oleh pengguna yang beralih ke `UpdateData` peran tersebut.

Selanjutnya, ubah izin untuk peran Pengembang.

Berikan akses ke peran tersebut

Pada titik ini, baik Analis dan Pengembang memiliki izin yang memungkinkan mereka mengelola data di akun Originating. Gunakan langkah-langkah berikut yang diperlukan untuk menambahkan izin agar dapat beralih ke peran.

Untuk memodifikasi peran Pengembang agar mereka dapat beralih ke UpdateData peran

1. Masuk sebagai administrator di akun Originating, dan buka IAM konsol.
2. Pilih Peran, lalu pilih Pengembang.
3. Pilih tab Permissions (Izin), pilih Add permissions (Tambahkan izin), lalu pilih Create inline policy (Membuat kebijakan inline).
4. Pilih JSONtab.
5. Tambahkan pernyataan kebijakan berikut untuk mengizinkan `AssumeRole` tindakan pada `UpdateData` peran di akun Tujuan. Pastikan bahwa Anda mengubah *`DESTINATION-ACCOUNT-ID`* dalam `Resource` elemen ke yang sebenarnya Akun AWS ID dari akun Tujuan.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::DESTINATION-ACCOUNT-ID:role/UpdateData"
     }
   }
   ```

   `Allow`Efeknya secara eksplisit memungkinkan grup Pengembang mengakses `UpdateData` peran di akun Tujuan. Setiap pengembang yang mencoba mengakses peran berhasil.
6. Pilih Tinjau kebijakan.
7. Ketik Nama seperti**`allow-assume-S3-role-in-destination`**.
8. Pilih Buat kebijakan.

Di sebagian besar lingkungan, Anda mungkin tidak memerlukan prosedur berikut. Namun, jika Anda menggunakan PowerUserAccess izin, maka beberapa grup mungkin sudah dapat beralih peran. Prosedur berikut menunjukkan cara menambahkan `"Deny"` izin ke grup Analis untuk memastikan bahwa mereka tidak dapat mengambil peran tersebut. Jika Anda tidak memerlukan prosedur ini di lingkungan Anda, maka kami sarankan Anda tidak menambahkannya. `"Deny"`izin membuat gambar izin keseluruhan lebih rumit untuk dikelola dan dipahami. Gunakan `"Deny"` izin hanya jika Anda tidak memiliki opsi yang lebih baik.

Untuk memodifikasi peran Analis untuk menolak izin untuk mengambil **`UpdateData`** peran

1. Pilih Peran, lalu pilih Analis.
2. Pilih tab Permissions (Izin), pilih Add permissions (Tambahkan izin), lalu pilih Create inline policy (Membuat kebijakan inline).
3. Pilih JSONtab.
4. Tambahkan pernyataan kebijakan berikut untuk menolak tindakan `AssumeRole` di peran `UpdateData`. Pastikan bahwa Anda mengubah *`DESTINATION-ACCOUNT-ID`* dalam `Resource` elemen ke yang sebenarnya Akun AWS ID dari akun Tujuan.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::DESTINATION-ACCOUNT-ID:role/UpdateData"
     }
   }
   ```

   `Deny`Efeknya secara eksplisit menyangkal akses grup Analis ke `UpdateData` peran dalam akun Tujuan. Setiap analis yang mencoba mengakses peran menerima pesan yang ditolak akses.
5. Pilih Tinjau kebijakan.
6. Ketik Nama seperti**`deny-assume-S3-role-in-destination`**.
7. Pilih Buat kebijakan.

Peran Pengembang sekarang memiliki izin untuk menggunakan `UpdateData` peran di akun Tujuan. Peran analis dicegah dari menggunakan `UpdateData` peran tersebut.

Selanjutnya, Anda dapat melihat bagaimana David, seorang pengembang, dapat mengakses `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. David dapat mengakses ember dari AWS Management Console, AWS CLI, atau AWS API.

Akses uji dengan mengalihkan peran

Setelah menyelesaikan dua langkah pertama tutorial ini, Anda memiliki peran yang memberikan akses ke sumber daya di akun Tujuan. Anda juga memiliki satu peran di akun Originating dengan pengguna diizinkan untuk menggunakan peran itu. Langkah ini membahas cara menguji beralih ke peran itu dari AWS Management Console, AWS CLI, dan AWS API.

Untuk mendapatkan bantuan dengan masalah umum yang mungkin Anda temui saat bekerja dengan IAM peran, lihatMemecahkan masalah peran IAM.

Beralih peran (konsol)

Jika David perlu memperbarui data di akun Tujuan di AWS Management Console, dia bisa melakukannya dengan menggunakan Switch Role. Dia menentukan ID akun atau alias dan nama peran, dan izinnya segera dialihkan kepada mereka yang diizinkan oleh peran tersebut. Dia kemudian dapat menggunakan konsol untuk bekerja dengan `amzn-s3-demo-bucket-shared-container` ember, tetapi tidak dapat bekerja dengan sumber daya lain di Destination. Sementara David menggunakan peran itu, dia juga tidak dapat memanfaatkan hak istimewa penggunanya di akun Originating. Itu karena hanya satu set izin yang dapat berlaku pada satu waktu.

IAMmenyediakan dua cara yang dapat digunakan David untuk masuk ke halaman Switch Role:

- David menerima tautan dari administrator mereka yang menunjuk ke konfigurasi Switch Role yang telah ditentukan sebelumnya. Tautan ini diberikan ke administrator di halaman terakhir dari panduan Buat peran atau pada halaman Ringkasan Peran untuk peran lintas akun. Memilih tautan ini membawa David ke halaman Alihkan Peran dengan bidang ID Akun dan Nama peran yang sudah diisi. Yang perlu dilakukan David adalah memilih Switch Roles.
- Administrator tidak mengirim tautan di surel, tetapi mengirim nilai ID Akun dan Nama Peran. Untuk beralih peran, David harus memasukkan nilai secara manual. Hal ini digambarkan dalam prosedur berikut.

Untuk mengambil peran

1. Daud menandatangani AWS Management Console menggunakan pengguna normalnya di akun Originating.

2. Mereka memilih tautan yang dikirimkan administrator melalui email kepada mereka. Ini membawa David ke halaman Beralih Peran dengan ID akun atau alias dan informasi nama peran yang sudah diisi.

   —atau—

   David memilih nama mereka (menu Identity) pada bilah navigasi, dan kemudian memilih Switch Roles.

   Jika ini adalah pertama kalinya David mencoba mengakses halaman Switch Role dengan cara ini, dia pertama kali mendarat di halaman Switch Role yang dijalankan pertama kali. Halaman ini memberikan informasi tambahan tentang bagaimana peralihan peran dapat memungkinkan pengguna untuk mengelola sumber daya di seluruh Akun AWS. David harus memilih Switch Role di halaman ini untuk menyelesaikan sisa prosedur ini.
3. Selanjutnya, untuk mengakses peran, David harus secara manual mengetikkan nomor ID akun Tujuan (`999999999999`) dan nama peran (`UpdateData`).

   Selain itu, David ingin memantau peran dan izin terkait yang saat ini aktif. IAM Untuk melacak informasi ini, dia mengetik `Destination` di kotak teks Display Nama (Nama Tampilan), memilih opsi warna merah, lalu memilih Switch Role (Alihkan Peran).
4. Sekarang, David dapat menggunakan konsol Amazon S3 untuk bekerja menggunakan bucket Amazon S3 atau sumber daya lain yang peran `UpdateData`nya memiliki izin.
5. Setelah selesai, David dapat kembali ke izin aslinya. Untuk melakukan itu, mereka memilih nama tampilan peran Tujuan pada bilah navigasi dan kemudian memilih Kembali ke David @ 1111111111.
6. Lain kali David ingin beralih peran dan memilih menu Identitas di bilah navigasi, dia melihat entri Tujuan masih ada dari terakhir kali. Dia cukup memilih entri itu untuk segera beralih peran tanpa memasukkan kembali ID akun dan nama peran.

Beralih peran (AWS CLI)

Jika David perlu bekerja di lingkungan Tujuan di baris perintah, dia dapat melakukannya dengan menggunakan AWS CLI. Dia menjalankan `aws sts assume-role` perintah dan meneruskan peran ARN untuk mendapatkan kredensil keamanan sementara untuk peran itu. Dia kemudian mengonfigurasi kredensil tersebut dalam variabel lingkungan jadi selanjutnya AWS CLI perintah bekerja menggunakan izin peran. Sementara David menggunakan peran tersebut, ia tidak dapat

menggunakan hak istimewa penggunanya di akun Originating, karena hanya satu set izin yang dapat berlaku pada satu waktu.

Perhatikan bahwa semua access key dan token hanyalah contoh dan tidak dapat digunakan seperti yang ditunjukkan. Ganti dengan nilai yang sesuai dari lingkungan langsung Anda.

Untuk mengambil peran

1. David membuka jendela prompt perintah, dan mengonfirmasi bahwa AWS CLI klien bekerja dengan menjalankan perintah:

   ```
   aws help
   ```

   > **Note**
   >
   > Lingkungan default David menggunakan kredensial pengguna `David` dari profil default yang ia buat dengan perintah `aws configure`. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS Command Line Interfacedi AWS Command Line Interface Panduan Pengguna.

2. Dia memulai proses peralihan peran dengan menjalankan perintah berikut untuk beralih ke `UpdateData` peran di akun Tujuan. Dia menerima peran ARN dari administrator yang menciptakan peran tersebut. Perintah ini mengharuskan Anda memberikan nama sesi, Anda juga dapat memilih teks apa pun yang Anda suka untuk itu.

   ```
   aws sts assume-role --role-arn "arn:aws:iam::999999999999:role/UpdateData" --role-
   session-name "David-ProdUpdate"
   ```

   David kemudian melihat hal-hal berikut di output:

   ```
   {
       "Credentials": {
           "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
           "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
   +Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
   CvSRyh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
   uZEXAMPLECihzFB5lTYLto9dyBgSDy
   EXAMPLE9/
   g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
   ```

```
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
        "Expiration": "2014-12-11T23:08:07Z",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

3. David melihat tiga bagian yang mereka butuhkan di bagian Credentials dari output.

   - `AccessKeyId`
   - `SecretAccessKey`
   - `SessionToken`

   David perlu mengkonfigurasi AWS CLI lingkungan untuk menggunakan parameter ini dalam panggilan berikutnya. Untuk informasi tentang berbagai cara mengonfigurasi kredensional Anda, lihat Mengonfigurasi AWS Command Line Interface. Anda tidak dapat menggunakan `aws configure` perintah karena tidak mendukung pengambilan token sesi. Namun, Anda dapat memasukkan informasi secara manual ke dalam file konfigurasi. Karena ini adalah kredensial sementara dengan waktu kedaluwarsa yang relatif singkat, paling mudah menambahkannya ke lingkungan sesi baris perintah Anda saat ini.

4. Untuk menambahkan tiga nilai ke lingkungan, David memotong dan menempelkan output langkah sebelumnya ke perintah berikutnya. Anda mungkin ingin memotong dan menempel ke dalam editor teks sederhana untuk mengatasi masalah line wrap dalam output token sesi. Ia harus ditambahkan sebagai satu string panjang, meskipun line wrap ditampilkan di sini untuk kejelasan.

   Contoh berikut menunjukkan perintah yang diberikan di lingkungan Windows, di mana “set” adalah perintah untuk membuat variabel lingkungan. Di komputer Linux atau macOS, Anda akan menggunakan perintah “ekspor” sebagai gantinya. Semua bagian contoh lainnya valid di ketiga lingkungan.

   Untuk detail tentang menggunakan Alat untuk Windows Powershell, lihat Beralih ke IAM peran (Alat untuk Windows PowerShell)

```
set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvS
```

```
Ryh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDyEXA
MPLEKEY9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UusKd
EXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLENhykxiHen
DHq6ikBQ==
```

Pada titik ini, setiap perintah berikut dijalankan di bawah izin peran yang diidentifikasi oleh kredensial tersebut. Dalam kasus David, peran `UpdateData`.

> **Important**
>
> Anda dapat menyimpan pengaturan konfigurasi dan kredensi yang sering digunakan dalam file yang dikelola oleh AWS CLI. Untuk informasi selengkapnya, lihat Menggunakan file konfigurasi dan kredensial yang ada di AWS Command Line Interface Panduan Pengguna.

5. Jalankan perintah untuk mengakses sumber daya di akun Tujuan. Dalam contoh ini, David mencantumkan isi bucket S3 mereka dengan perintah berikut.

```
aws s3 ls s3://shared-container
```

Karena nama bucket Amazon S3 unik secara universal, tidak perlu menentukan ID akun yang memiliki bucket. Untuk mengakses sumber daya untuk orang lain AWS layanan, lihat AWS CLI dokumentasi untuk layanan itu untuk perintah dan sintaks yang diperlukan untuk mereferensikan sumber dayanya.

Menggunakan AssumeRole (AWS API)

Ketika David perlu membuat pembaruan ke akun Tujuan dari kode, dia membuat `AssumeRole` panggilan untuk mengambil `UpdateData` peran. Panggilan mengembalikan kredensi sementara yang dapat dia gunakan untuk mengakses `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. Dengan kredensi tersebut, David dapat melakukan API panggilan untuk memperbarui ember. `amzn-s3-demo-bucket-shared-container` Namun, ia tidak dapat melakukan API panggilan untuk mengakses sumber daya lain di akun Tujuan, meskipun ia memiliki izin pengguna daya di akun Originating.

Untuk mengambil peran

1. David memanggil `AssumeRole` sebagai bagian dari aplikasi. Mereka harus menentukan `UpdateDataARN:arn:aws:iam::999999999999:role/UpdateData`.

   Respond dari panggilan `AssumeRole` mencakup kredensial sementara dengan `AccessKeyId` dan `SecretAccessKey`. Informasi ini juga mencakup waktu `Expiration` yang menunjukkan kapan kredensial kedaluwarsa dan Anda harus meminta yang baru. Saat Anda mengatur rantai peran dengan AWS SDK, banyak penyedia kredensi secara otomatis menyegarkan kredensil sebelum kedaluwarsa.
2. Dengan kredensial sementara, David membuat panggilan `s3:PutObject` untuk memperbarui bucket `amzn-s3-demo-bucket-shared-container`. Mereka akan meneruskan kredensil ke API panggilan sebagai parameter. `AuthParams` Karena kredenal peran sementara hanya memiliki akses baca dan tulis ke `amzn-s3-demo-bucket-shared-container` bucket, tindakan lain apa pun di akun Tujuan ditolak.

Untuk contoh kode (menggunakan Python), lihat Beralih ke IAM peran (AWS API).

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang topik dalam tutorial ini:

- Untuk informasi selengkapnya tentang IAM pengguna, lihatIAMIdentitas .
- Untuk informasi selengkapnya tentang bucket Amazon S3, lihat Membuat Bucket di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.
- Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? .

Ringkasan

Anda telah menyelesaikan tutorial API akses lintas akun. Anda menciptakan peran untuk membangun kepercayaan dengan akun lainnya dan menetapkan tindakan apa yang dapat dilakukan oleh entitas tepercaya. Kemudian, Anda mengubah kebijakan peran untuk mengontrol IAM pengguna mana yang dapat mengakses peran tersebut. Akibatnya, pengembang dari akun Originating dapat melakukan pembaruan ke `amzn-s3-demo-bucket-shared-container` bucket di akun Destination dengan menggunakan kredensi sementara.

IAMtutorial: Buat dan lampirkan kebijakan terkelola pelanggan pertama Anda

Dalam tutorial ini, Anda menggunakan AWS Management Console untuk membuat kebijakan yang dikelola pelanggan dan kemudian melampirkan kebijakan itu ke IAM pengguna di Anda Akun AWS. Kebijakan yang Anda buat memungkinkan pengguna IAM pengujian untuk masuk langsung ke izin AWS Management Console hanya-baca.

Alur kerja ini memiliki tiga langkah dasar:

Langkah 1: Buat kebijakan

Secara default, IAM pengguna tidak memiliki izin untuk melakukan apa pun. Mereka tidak dapat mengakses AWS Management Console atau mengelola data di dalamnya, kecuali jika Anda mengizinkannya. Pada langkah ini, Anda membuat kebijakan yerkelola pelanggan yang memungkinkan setiap pengguna yang melekat untuk masuk ke konsol.

Langkah 2: Lampirkan kebijakan

Saat Anda melampirkan kebijakan kepada pengguna, pengguna menurunkan semua izin akses yang terkait dengan kebijakan tersebut. Pada langkah ini, Anda melampirkan kebijakan baru ke pengguna pengujian.

Langkah 3: Uji akses pengguna

Setelah kebijakan tersebut dilampirkan, Anda dapat masuk sebagai pengguna dan menguji kebijakan tersebut.

Prasyarat

Untuk melakukan langkah-langkah di tutorial ini, Anda harus sudah memiliki hal-hal berikut:

- Sebuah Akun AWS yang dapat Anda masuki sebagai IAM pengguna dengan izin administratif.
- IAMPengguna uji yang tidak memiliki izin yang ditetapkan atau keanggotaan grup sebagai berikut:

Nama pengguna	Grup	Izin
PolicyUser	<tidak ada>	<tidak ada>

Langkah 1: Buat kebijakan

Pada langkah ini, Anda membuat kebijakan terkelola pelanggan yang memungkinkan pengguna terlampir masuk AWS Management Console dengan akses data hanya-baca. IAM

Untuk membuat kebijakan bagi pengguna uji Anda

1. Masuk ke IAM konsol di https://console.aws.amazon.com/iam/dengan pengguna Anda yang memiliki izin administrator.
2. Di panel navigasi, pilih Kebijakan.
3. Di panel konten, pilih Buat kebijakan.
4. Pilih JSONopsi dan salin teks dari dokumen JSON kebijakan berikut. Tempelkan teks ini ke dalam kotak JSONteks.

```
{
    "Version": "2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    } ]
}
```

5. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih kebijakan Tinjauan di tab Editor visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

6. Pada halaman Tinjau dan buat, ketik **`UsersReadOnlyAccessToIAMConsole`** nama kebijakan. Tinjau izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul di daftar kebijakan terkelola dan siap dilampirkan.

Langkah 2: Lampirkan kebijakan

Selanjutnya Anda melampirkan kebijakan yang baru saja Anda buat untuk IAM pengguna pengujian Anda.

Untuk melampirkan kebijakan ke pengguna uji Anda

1. Di IAM konsol, di panel navigasi, pilih Kebijakan.
2. Di bagian atas daftar kebijakan, di kotak pencarian, mulailah mengetik **`UsersReadOnlyAccesstoIAMConsole`** hingga Anda dapat melihat kebijakan Anda. Kemudian pilih tombol radio UsersReadOnlyAccessToIAMConsoledi sebelah dalam daftar.
3. Pilih tombol Tindakan, lalu pilih Lampirkan.
4. Di IAM entitas pilih opsi untuk memfilter untuk Pengguna.
5. Di kotak pencarian, mulailah mengetik **`PolicyUser`** hingga pengguna terlihat di daftar. Lalu centang kotak di samping pengguna tersebut di daftar.
6. Pilih Lampirkan kebijakan.

Anda telah melampirkan kebijakan ke pengguna IAM pengujian Anda, yang berarti pengguna sekarang memiliki akses hanya-baca ke konsol. IAM

Langkah 3: Uji akses pengguna

Untuk tutorial ini, kami sarankan agar Anda menguji akses dengan masuk sebagai pengguna uji sehingga Anda dapat melihat apa yang pengguna mungkin alami.

Untuk menguji akses dengan masuk dengan pengguna pengujian

1. Masuk ke IAM konsol di https://console.aws.amazon.com/iam/dengan pengguna `PolicyUser` pengujian Anda.

2. Jelajahi halaman konsol dan coba buat pengguna atau grup baru. Perhatikan bahwa `PolicyUser` dapat menampilkan data tetapi tidak dapat membuat atau memodifikasi IAM data yang ada.

Sumber daya terkait

Untuk informasi terkait, lihat sumber daya berikut:

- Kebijakan terkelola dan kebijakan inline
- Kontrol akses IAM pengguna ke AWS Management Console

Ringkasan

Anda telah berhasil menyelesaikan semua langkah yang diperlukan untuk membuat dan melampirkan kebijakan terkelola pelanggan. Akibatnya, Anda dapat masuk ke IAM konsol dengan akun pengujian Anda untuk melihat seperti apa pengalaman itu bagi pengguna Anda.

IAMtutorial: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag

Attribute-based access control (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke IAM sumber daya, termasuk IAM entitas (pengguna atau peran) dan AWS sumber daya. Kemudian, Anda dapat menentukan kebijakan yang menggunakan kunci syarat tanda untuk memberikan izin kepada penanggung jawab Anda berdasarkan tanda mereka. Saat Anda menggunakan tag untuk mengontrol akses ke AWS sumber daya Anda, Anda memungkinkan tim dan sumber daya Anda tumbuh dengan lebih sedikit perubahan pada AWS kebijakan. ABACKebijakan lebih fleksibel daripada AWS kebijakan tradisional, yang mengharuskan Anda untuk mencantumkan setiap sumber daya individu. Untuk informasi lebih lanjut tentang ABAC dan keunggulannya dibandingkan kebijakan tradisional, lihatTentukan izin berdasarkan atribut dengan otorisasi ABAC.

> **Note**
>
> Anda harus memberikan satu nilai untuk setiap tag sesi. AWS Security Token Service tidak mendukung tag sesi multi-nilai.

Topik

Gambaran umum tutorial

Tutorial ini menunjukkan cara membuat dan menguji kebijakan yang memungkinkan IAM peran dengan tag utama untuk mengakses sumber daya dengan tag yang cocok. Saat prinsipal mengajukan permintaan ke AWS, izin mereka diberikan berdasarkan apakah tanda prinsipal dan tanda sumber daya cocok. Strategi ini memungkinkan individu untuk melihat atau mengedit hanya AWS sumber daya yang diperlukan untuk pekerjaan mereka.

Skenario

Asumsikan bahwa Anda adalah pengembang utama di sebuah perusahaan besar bernama Example Corporation, dan Anda adalah IAM administrator yang berpengalaman. Anda terbiasa membuat dan mengelola IAM pengguna, peran, dan kebijakan. Anda ingin memastikan anggota teknisi pengembangan dan tim jaminan kualitas Anda dapat mengakses sumber daya yang mereka butuhkan. Anda juga memerlukan strategi yang dapat menskalakan pertumbuhan perusahaan Anda.

Anda memilih untuk menggunakan tag AWS sumber daya dan tag utama IAM peran untuk menerapkan ABAC strategi untuk layanan yang mendukungnya, dimulai dengan AWS Secrets Manager. Untuk mempelajari layanan mana yang mendukung otorisasi berdasarkan tanda, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari kunci kondisi penandaan yang dapat Anda gunakan dalam kebijakan dengan setiap tindakan dan sumber daya layanan, lihat Tindakan,

Sumber Daya, dan Kunci Kondisi untuk AWS Layanan. Anda dapat mengonfigurasi penyedia identitas SAML berbasis atau web Anda untuk meneruskan tag sesi AWS. Ketika karyawan Anda bergabung AWS, atribut mereka diterapkan pada AWS prinsipal yang dihasilkan. Anda kemudian dapat menggunakan ABAC untuk mengizinkan atau menolak izin berdasarkan atribut tersebut. Untuk mempelajari bagaimana menggunakan tag sesi dengan identitas SAML federasi berbeda dari tutorial ini, lihatIAMtutorial: Gunakan tag SAML sesi untuk ABAC.

Anggota tim Insinyur dan Jaminan Kualitas Anda berada di proyek Pegasus atau proyekUnicorn Anda. Anda memilih proyek 3 karakter dan nilai tanda tim berikut:

- `access-project` = `peg` untuk proyek Pegasus
- `access-project` = `uni` untuk proyek Unicorn
- `access-team` = `eng` untuk tim Insinyur
- `access-team` = `qas` untuk tim Jaminan Kualitas

Selain itu, Anda memilih untuk meminta tag alokasi `cost-center` biaya untuk mengaktifkan laporan AWS penagihan kustom. Untuk informasi selengkapnya, lihat Penggunaan Tanda Alokasi Biaya dalam Panduan Pengguna AWS Billing and Cost Management .

Ringkasan keputusan penting

- Karyawan masuk dengan kredensi IAM pengguna dan kemudian mengambil IAM peran untuk tim dan proyek mereka. Jika perusahaan Anda memiliki sistem identitas sendiri, Anda dapat mengatur federasi untuk memungkinkan karyawan untuk mengambil peran tanpa IAM pengguna. Untuk informasi selengkapnya, lihat IAMtutorial: Gunakan tag SAML sesi untuk ABAC.
- Kebijakan yang sama terlampir pada semua peran. Tindakan diizinkan atau ditolak berdasarkan tanda.
- Karyawan dapat membuat sumber daya baru, tetapi hanya jika mereka melampirkan tanda yang sama ke sumber daya yang diterapkan pada peran mereka. Hal ini memastikan bahwa karyawan dapat melihat sumber daya setelah mereka membuatnya. Administrator tidak lagi diharuskan memperbarui kebijakan dengan sumber ARN daya baru.
- Karyawan dapat membaca sumber daya yang dimiliki oleh tim mereka, apa pun proyeknya.
- Karyawan dapat memperbarui dan menghapus sumber daya yang dimiliki oleh tim dan proyek mereka sendiri.

- IAMAdministrator dapat menambahkan peran baru untuk proyek baru. Mereka dapat membuat dan menandai IAM pengguna baru untuk memungkinkan akses ke peran yang sesuai. Administrator tidak diwajibkan untuk mengedit kebijakan untuk mendukung proyek atau anggota tim baru.

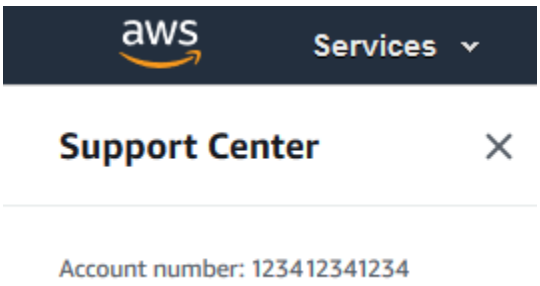
Di tutorial ini, Anda akan menandai setiap sumber daya, menandai peran proyek Anda, dan menambahkan kebijakan ke peran untuk memungkinkan perilaku yang dijelaskan sebelumnya. Kebijakan yang dihasilkan memungkinkan peran `Create`, `Read`, `Update`, dan `Delete` mengakses sumber daya yang ditandai dengan tanda proyek dan tanda tim yang sama. Kebijakan ini juga memungkinkan proyek silang `Read` untuk mengakses sumber daya yang ditandai dengan tim yang sama.

Prasyarat

Untuk melakukan langkah-langkah di tutorial ini, Anda harus sudah memiliki hal-hal berikut:

- Sebuah Akun AWS yang dapat Anda masuki sebagai pengguna dengan izin administratif.
- ID akun 12 digit Anda, yang Anda gunakan untuk membuat peran di langkah 3.

  Untuk menemukan nomor ID AWS akun Anda menggunakan AWS Management Console, pilih Support pada bilah navigasi di kanan atas, lalu pilih Support Center. Nomor akun (ID) muncul di panel navigasi di sebelah kiri.



- Pengalaman membuat dan mengedit IAM pengguna, peran, dan kebijakan di AWS Management Console. Namun, jika Anda memerlukan bantuan mengingat proses IAM manajemen, tutorial ini menyediakan tautan di mana Anda dapat melihat step-by-step instruksi.

Langkah 1: Buat pengguna uji

Untuk pengujian, buat empat IAM pengguna dengan izin untuk mengambil peran dengan tag yang sama. Ini memudahkan untuk menambahkan lebih banyak pengguna ke tim Anda. Saat Anda menandai pengguna, mereka secara otomatis mendapatkan akses untuk mengambil peran yang

tepat. Anda tidak perlu menambahkan pengguna ke kebijakan kepercayaan peran tersebut jika mereka hanya menangani satu proyek dan tim.

1. Buat kebijakan yang dikelola pelanggan berikut bernama `access-assume-role`. Untuk informasi selengkapnya tentang membuat JSON kebijakan, lihatMembuat IAM kebijakan.

   ABACpolicy: Asumsikan ABAC peran apa pun, tetapi hanya jika pengguna dan tag peran cocok

   Kebijakan berikut memungkinkan pengguna mengambil peran apa pun di akun Anda dengan prefiks nama `access-`. Peran tersebut juga harus ditandai dengan tanda proyek, tim, dan pusat biaya yang sama dengan pengguna.

   Untuk menggunakan kebijakan ini, ganti teks placeholder yang dicetak miring dengan informasi akun Anda.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "TutorialAssumeRole",
               "Effect": "Allow",
               "Action": "sts:AssumeRole",
               "Resource": "arn:aws:iam::account-ID-without-hyphens:role/access-*",
               "Condition": {
                   "StringEquals": {
                       "iam:ResourceTag/access-project": "${aws:PrincipalTag/access-
   project}",
                       "iam:ResourceTag/access-team": "${aws:PrincipalTag/access-
   team}",
                       "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
   center}"
                   }
               }
           }
       ]
   }
   ```

   Untuk menskalakan tutorial ini ke sejumlah besar pengguna, Anda dapat melampirkan kebijakan ke grup dan menambahkan setiap pengguna ke grup. Untuk informasi selengkapnya, silakan lihat Buat grup IAM pengguna dan Mengedit pengguna dalam IAM IAM grup.

2. Buat IAM pengguna berikut, lampirkan kebijakan `access-assume-role` izin. Pastikan Anda memilih Menyediakan akses pengguna ke AWS Management Console, dan kemudian menambahkan tag berikut.

Nama pengguna	Kunci tag pengguna	Nilai tag pengguna
Akses-a rnav-peg-eng	`access-project`	`peg`
	`access-team`	`eng`
	`cost-center`	`987654`
Akses-m ary-peg-qas	`access-project`	`peg`
	`access-team`	`qas`
	`cost-center`	`987654`
Akses-s aanvi-uni-eng	`access-project`	`uni`
	`access-team`	`eng`
	`cost-center`	`123456`
Akses-c arlos-uni-qas	`access-project`	`uni`
	`access-team`	`qas`
	`cost-center`	`123456`

Langkah 2: Buat ABAC kebijakan

Buat kebijakan berikut bernama **`access-same-project-team`**. Anda akan menambahkan kebijakan ini ke peran tersebut pada langkah berikutnya. Untuk informasi selengkapnya tentang membuat JSON kebijakan, lihatMembuat IAM kebijakan.

Untuk kebijakan tambahan yang dapat Anda sesuaikan untuk tutorial ini, lihat halaman berikut:

- Mengontrol akses untuk prinsipal IAM

- Amazon EC2: Memungkinkan memulai atau menghentikan instans EC2 yang telah ditandai pengguna, secara terprogram, dan di konsol
- EC2: Mulai atau hentikan instans berdasarkan pada pencocokan prinsipal dan tanda sumber daya
- EC2: Mulai atau hentikan instans berdasarkan tanda
- IAM: Asumsikan peran yang memiliki tag tertentu

ABACKebijakan: Akses Sumber Daya Secrets Manager Hanya Saat Prinsipal dan Tag Sumber Daya Cocokkan

Kebijakan berikut memungkinkan penanggung jawab membuat, membaca, mengedit, dan menghapus sumber daya, tetapi hanya jika sumber daya tersebut ditandai dengan pasangan nilai kunci yang sama dengan penanggung jawab. Saat prinsipal membuat sumber daya, mereka harus menambahkan tanda `access-project`, `access-team`, dan `cost-center` dengan nilai yang cocok dengan tanda prinsipal. Kebijakan juga memungkinkan penambahan opsi tanda `Name` atau `OwnedBy`.

```
{
 "Version": "2012-10-17",
 "Statement": [
     {
         "Sid": "AllActionsSecretsManagerSameProjectSameTeam",
         "Effect": "Allow",
         "Action": "secretsmanager:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
                 "aws:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                 "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
             },
             "ForAllValues:StringEquals": {
                 "aws:TagKeys": [
                     "access-project",
                     "access-team",
                     "cost-center",
                     "Name",
                     "OwnedBy"
                 ]
             },
```

```
            "StringEqualsIfExists": {
                "aws:RequestTag/access-project": "${aws:PrincipalTag/access-project}",
                "aws:RequestTag/access-team": "${aws:PrincipalTag/access-team}",
                "aws:RequestTag/cost-center": "${aws:PrincipalTag/cost-center}"
            }
        }
    },
    {
        "Sid": "AllResourcesSecretsManagerNoTags",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:GetRandomPassword",
            "secretsmanager:ListSecrets"
        ],
        "Resource": "*"
    },
    {
        "Sid": "ReadSecretsManagerSameTeam",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:Describe*",
            "secretsmanager:Get*",
            "secretsmanager:List*"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
            }
        }
    },
    {
        "Sid": "DenyUntagSecretsManagerReservedTags",
        "Effect": "Deny",
        "Action": "secretsmanager:UntagResource",
        "Resource": "*",
        "Condition": {
            "ForAnyValue:StringLike": {
                "aws:TagKeys": "access-*"
            }
        }
    },
    {
        "Sid": "DenyPermissionsManagement",
```

```
        "Effect": "Deny",
        "Action": "secretsmanager:*Policy",
        "Resource": "*"
    }
 ]
}
```

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllActionsSecretsManagerSameProjectSameTeam` memungkinkan semua tindakan layanan ini pada semua sumber daya terkait, tetapi hanya jika tanda sumber daya cocok dengan tanda prinsipal. Dengan menambahkan `"Action": "secretsmanager:*"` ke kebijakan, akan menumbuhkan kebijakan seiring bertumbuhnya Secrets Manager. Jika Secrets Manager menambahkan API operasi baru, Anda tidak diharuskan menambahkan tindakan itu ke pernyataan. Pernyataan tersebut mengimplementasikan ABAC menggunakan tiga blok kondisi. Permintaan hanya diperbolehkan jika ketiga blok dikembalikan dengan benar.
  - Blok kondisi pertama dari pernyataan ini dikembalikan dengan benar jika kunci tanda yang ditentukan ada pada sumber daya, dan nilainya cocok dengan tanda prinsipal. Blok ini mengembalikan kesalahan tanda yang tidak sesuai, atau tindakan yang tidak mendukung penandaan sumber daya. Untuk mempelajari tindakan mana yang tidak diizinkan oleh blok ini, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Secrets Manager. Halaman tersebut menunjukkan bahwa tindakan yang dilakukan pada tipe sumber daya Rahasia mendukung `secretsmanager:ResourceTag/tag-key` kunci kondisi. Beberapa tindakan Secrets Manager tidak mendukung tipe sumber daya tersebut, termasuk `GetRandomPassword` dan `ListSecrets`. Anda harus membuat pernyataan tambahan untuk mengizinkan tindakan tersebut.
  - Blok kondisi kedua menjadi benar jika setiap kunci tanda yang diteruskan di permintaan disertakan dalam daftar yang ditentukan. Hal ini dilakukan menggunakan `ForAllValues` dengan operator kondisi `StringEquals`. Jika tidak ada kunci atau subset dari set kunci yang diteruskan, kondisi menjadi benar. Hal ini memungkinkan operasi `Get*` yang tidak memungkinkan tanda diteruskan di permintaan. Jika pemohon menyertakan kunci tanda yang tidak ada dalam daftar, kondisi menjadi salah. Setiap kunci tanda yang diteruskan di permintaan harus sesuai dengan anggota daftar ini. Untuk informasi selengkapnya, lihat Kunci konteks multivaluasi.
  - Blok kondisi ketiga menjadi benar jika permintaan mendukung tanda yang lolos, jika ketiga tanda hadir, dan jika sesuai dengan nilai tanda prinsipal. Blok ini juga menjadi benar jika permintaan tidak mendukung tanda yang diteruskan. Ini berkat `...IfExists` di operator kondisi. Blok

menjadi salah jika tidak ada tanda yang diteruskan selama tindakan yang mendukungnya, atau jika kunci dan nilai tanda tidak cocok.

- Pernyataan `AllResourcesSecretsManagerNoTags` memungkinkan tindakan `GetRandomPassword` dan `ListSecrets` yang tidak diperbolehkan oleh pernyataan pertama.
- Pernyataan `ReadSecretsManagerSameTeam` memungkinkan operasi hanya baca jika prinsipal ditandai dengan tanda access-team yang sama sebagai sumber daya. Hal ini diperbolehkan terlepas dari tanda proyek atau tanda pusat biaya.
- Pernyataan `DenyUntagSecretsManagerReservedTags` menolak permintaan untuk menghapus tanda dengan kunci yang dimulai dengan "access-" dari Secrets Manager. Tanda ini digunakan untuk mengontrol akses ke sumber daya, sehingga menghapus tanda dapat menghapus izin.
- Pernyataan `DenyPermissionsManagement` menolak akses untuk membuat, mengedit, atau menghapus kebijakan berbasis sumber daya Secrets Manager. Kebijakan ini dapat digunakan untuk mengubah izin rahasia.

> **Important**
>
> Kebijakan ini menggunakan strategi untuk memungkinkan semua tindakan untuk layanan, tetapi dengan tegas menolak tindakan yang melanggar izin. Menolak tindakan akan membatalkan kebijakan lain yang memungkinkan penanggung jawab melakukan tindakan tersebut. Ini dapat menimbulkan hasil yang tidak diinginkan. Sebagai praktik terbaik, gunakanlah penolakan eksplisit hanya jika tidak ada keadaan yang mengizinkan tindakan tersebut. Jika tidak, izinkan daftar tindakan individu, dan tindakan yang tidak diinginkan ditolak secara default.

Langkah 3: Buat peran

Buat IAM peran berikut dan lampirkan **`access-same-project-team`** kebijakan yang Anda buat di langkah sebelumnya. Untuk informasi selengkapnya tentang membuat IAM peran, lihatMembuat peran untuk mendelegasikan izin ke pengguna IAM. Jika Anda memilih untuk menggunakan federasi alih-alih IAM pengguna dan peran, lihatIAMtutorial: Gunakan tag SAML sesi untuk ABAC.

Fungsi pekerjaan	Nama peran	Tanda peran	Deskripsi peran
Proyek Teknik Pegasus	access-peg-engineering	akses-proyek = peg	Memungkinkan insinyur

Fungsi pekerjaan	Nama peran	Tanda peran	Deskripsi peran
		tim akses = `eng` pusat biaya = `987654`	membaca semua sumber daya teknik dan membuat serta mengelola sumber daya teknik Pegasus.
Jaminan Kualitas Proyek Pegasus	access-peg-quality-assurance	akses-proyek = `peg` tim akses = `qas` pusat biaya = `987654`	Memungkinkan tim QA untuk membaca semua sumber daya QA dan membuat, serta mengelola semua sumber daya QA Pegasus.
Proyek Teknik Unicorn	access-uni-engineering	akses-proyek = `uni` tim akses = `eng` pusat biaya = `123456`	Memungkinkan insinyur membaca semua sumber daya teknik dan membuat, serta mengelola sumber daya teknik Unicorn.

Fungsi pekerjaan	Nama peran	Tanda peran	Deskripsi peran
Jaminan Kualitas Proyek Unicorn	access-uni-quality-assurance	akses-proyek = `uni` tim akses = `qas` pusat biaya = `123456`	Memungkinkan tim QA untuk membaca semua sumber daya QA dan membuat, serta mengelola semua sumber daya QA Unicorn.

Langkah 4: Uji pembuatan rahasia

Kebijakan izin yang terlampir pada peran memungkinkan karyawan untuk membuat rahasia. Hal ini hanya diperbolehkan jika rahasia ditandai dengan proyek, tim, dan pusat biaya mereka. Konfirmasikan bahwa izin Anda bekerja sesuai harapan dengan masuk sebagai pengguna Anda, dengan asumsi peran yang benar, dan menguji aktivitas di Secrets Manager.

Untuk mencoba membuat rahasia dengan dan tanpa tag yang diperlukan

1. Di jendela browser utama Anda, tetap masuk sebagai pengguna administrator sehingga Anda dapat meninjau pengguna, peran, dan kebijakanIAM. Gunakan jendela penyamaran peramban atau peramban terpisah untuk pengujian Anda. Di sana, masuk sebagai `access-Arnav-peg-eng` IAM pengguna dan buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Upayakan untuk beralih ke peran `access-uni-engineering`.

   Operasi ini gagal karena nilai tanda `access-project` dan `cost-center` tidak cocok dengan pengguna `access-Arnav-peg-eng` dan peran `access-uni-engineering`.

   Untuk informasi selengkapnya tentang beralih peran di AWS Management Console, lihat Beralih dari pengguna ke IAM peran (konsol)
3. Beralih ke peran `access-peg-engineering`.
4. Simpan rahasia baru menggunakan informasi berikut. Untuk mempelajari cara menyimpan rahasia, lihat Membuat Rahasia Dasar di Panduan Pengguna AWS Secrets Manager .

> **Important**
>
> Secrets Manager menampilkan peringatan bahwa Anda tidak memiliki izin untuk AWS layanan tambahan yang bekerja dengan Secrets Manager. Misalnya, untuk membuat kredensil untuk RDS database Amazon, Anda harus memiliki izin untuk mendeskripsikan RDS instans, RDS klaster, dan klaster Amazon Redshift. Anda dapat mengabaikan peringatan ini karena Anda tidak menggunakan AWS layanan khusus ini dalam tutorial ini.

1. Di bagian Pilih tipe rahasia, pilih Tipe rahasia lainnya. Di dua kotak teks, masukkan `test-access-key` dan `test-access-secret`.
2. Masukkan `test-access-peg-eng` untuk bidang Nama rahasia.
3. Tambahkan kombinasi tanda yang berbeda dari tabel berikut dan lihat perilaku yang diharapkan.
4. Pilih Toko untuk mencoba membuat rahasia. Jika penyimpanan gagal, kembali ke laman konsol Secrets Manager sebelumnya dan gunakan set tag berikutnya dari tabel berikut. Set tag terakhir dizinkan dan akan berhasil membuat rahasia.

Tabel berikut menunjukkan kombinasi ABAC tag untuk `test-access-peg-eng` peran tersebut.

Nilai Tanda **`access-project`**	Nilai Tanda **`access-team`**	Nilai Tanda **`cost-center`**	Tanda tambahan	Perilaku yang diharapkan
(tidak ada)	(tidak ada)	(tidak ada)	(tidak ada)	Ditolak karena nilai tag `access-project` tidak cocok dengan nilai peran `peg`.
`uni`	`eng`	`987654`	(tidak ada)	Ditolak karena nilai tag `access-project` tidak cocok dengan nilai peran `peg`.

Nilai Tanda **access-project**	Nilai Tanda **access-team**	Nilai Tanda **cost-center**	Tanda tambahan	Perilaku yang diharapkan
peg	qas	987654	(tidak ada)	Ditolak karena nilai tag access-team tidak cocok dengan nilai peran eng.
peg	eng	123456	(tidak ada)	Ditolak karena nilai tag cost-center tidak cocok dengan nilai peran 987654.
peg	eng	987654	Pemilik = Jane	Ditolak karena tag tambahan owner tidak diizinkan oleh kebijakan, meskipun ketiga tanda yang diperlukan hadir dan nilainya sesuai dengan nilai-nilai peran.
peg	eng	987654	Nama = Jane	Diizinkan karena ketiga tanda yang diperlukan ada dan nilainya sesuai dengan nilai-nilai peran. Anda juga diperbolehkan untuk menyertakan tanda Name.

5. Keluar dan ulangi tiga langkah pertama dalam prosedur ini untuk setiap peran dan nilai tanda berikut. Pada langkah keempat dalam prosedur ini, uji setiap set tanda yang hilang, tanda opsional, tanda yang tidak diizinkan, dan nilai tanda tidak valid yang Anda pilih. Kemudian gunakan tanda yang diperlukan untuk membuat rahasia dengan tanda dan nama berikut.

Nama pengguna	Nama peran	Nama rahasia	Tanda rahasia
Akses-m ary-peg-q as	access-peg-quality-assurance	test-access-peg-qas	akses-proyek = peg tim akses = qas pusat biaya = 987654

Nama pengguna	Nama peran	Nama rahasia	Tanda rahasia
Akses-s aanvi-uni-eng	access-uni-engineering	test-access-uni-eng	akses-proyek = `uni` tim akses = `eng` pusat biaya = `123456`
Akses-c arlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	akses-proyek = `uni` tim akses = `qas` pusat biaya = `123456`

Langkah 5: Uji menampilkan rahasia

Kebijakan yang Anda lampirkan ke setiap peran memungkinkan karyawan melihat rahasia apa pun yang ditandai dengan nama tim mereka, apa pun proyek mereka. Konfirmasikan bahwa izin Anda bekerja sesuai harapan dengan menguji peran Anda di Secrets Manager.

Untuk uji melihat rahasia dengan dan tanpa tag yang diperlukan

1. Masuk sebagai salah satu IAM pengguna berikut:
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
2. Beralih ke peran yang sesuai:
   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-uni-quality-assurance`

Untuk informasi selengkapnya tentang beralih peran di AWS Management Console, lihatBeralih dari pengguna ke IAM peran (konsol).

3. Di panel navigasi sebelah kiri, pilih ikon menu untuk memperluas menu lalu pilih Rahasia.
4. Anda akan melihat keempat rahasia di tabel, apa pun peran Anda saat ini. Ini diharapkan karena kebijakan bernama `access-same-project-team` memungkinkan tindakan `secretsmanager:ListSecrets` untuk semua sumber daya.
5. Pilih nama salah satu rahasia.
6. Pada halaman detail rahasia, tanda peran Anda menentukan apakah Anda dapat melihat isi halaman tersebut. Bandingkan nama peran Anda dengan nama rahasia Anda. Jika mereka memiliki nama tim yang sama, tanda `access-team` cocok. Jika tidak cocok, akses akan ditolak.

   Tabel berikut menunjukkan perilaku melihat ABAC rahasia untuk setiap peran.

Nama peran	Nama rahasia	Perilaku yang diharapkan
`access-peg-engineering`	`test-access-peg-eng`	Diizinkan
	test-access-peg-qas	Ditolak
	test-access-uni-eng	Diizinkan
	test-access-uni-qas	Ditolak
access-peg-quality-assurance	test-access-peg-eng	Ditolak
	test-access-peg-qas	Diizinkan
	test-access-uni-eng	Ditolak
	test-access-uni-qas	Diizinkan
access-uni-engineering	test-access-peg-eng	Diizinkan
	test-access-peg-qas	Ditolak
	test-access-uni-eng	Diizinkan
	test-access-uni-qas	Ditolak

Nama peran	Nama rahasia	Perilaku yang diharapkan
access-uni-quality-assurance	test-access-peg-eng	Ditolak
	test-access-peg-qas	Diizinkan
	test-access-uni-eng	Ditolak
	test-access-uni-qas	Diizinkan

7. Dari breadcrumbs di bagian atas halaman, pilih Rahasia untuk kembali ke daftar rahasia. Ulangi langkah-langkah dalam prosedur ini menggunakan peran yang berbeda untuk menguji apakah Anda dapat melihat masing-masing rahasia.

Langkah 6: Uji skalabilitas

Alasan penting untuk menggunakan kontrol akses berbasis atribut (ABAC) atas kontrol akses berbasis peran () adalah skalabilitas. RBAC Saat perusahaan Anda menambahkan proyek, tim, atau orang baru AWS, Anda tidak perlu memperbarui kebijakan yang ABAC digerakkan oleh Anda. Misalnya, anggaplah bahwa Example Corporation mendanai proyek baru dengan nama kode Centaur. Seorang insinyur bernama Saanvi Sarkar akan menjadi insinyur utama untuk Centaur sambil terus mengerjakan proyek Unicorn. Saanvi juga akan meninjau pekerjaan untuk proyek Peg. Ada juga beberapa insinyur yang baru direkrut, termasuk Nikhil Jayashankar, yang hanya akan mengerjakan proyek Centaur.

Untuk menambahkan proyek baru ke AWS

1. Masuk sebagai pengguna IAM administrator dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Peran dan tambahkan IAM peran bernama`access-cen-engineering`. Lampirkan kebijakan **`access-same-project-team`** izin ke peran dan tambahkan tag peran berikut:
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
3. Pada panel navigasi di sebelah kiri, pilih Pengguna.

4. Tambahkan pengguna baru bernama`access-Nikhil-cen-eng`, lampirkan kebijakan bernama`access-assume-role`, dan tambahkan tag pengguna berikut.

   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`

5. Gunakan prosedur di Langkah 4: Uji pembuatan rahasia dan Langkah 5: Uji menampilkan rahasia. Di jendela peramban lain, uji apakah Nikhil hanya dapat membuat rahasia teknik Centaur, dan apakah dia dapat melihat semua rahasia teknik.

6. Di jendela browser utama tempat Anda masuk sebagai administrator, pilih penggguna`access-Saanvi-uni-eng`.

7. Pada tab Izin, hapus kebijakan access-assume-roleizin.

8. Tambahkan kebijakan inline berikut bernama `access-assume-specific-roles`. Untuk informasi selengkapnya tentang menambahkan kebijakan inline ke pengguna, lihat Untuk menyematkan kebijakan inline bagi pengguna atau peran (konsole).

   ABACkebijakan: Asumsikan hanya peran tertentu

   Kebijakan ini memungkinkan Saanvi untuk mengambil peran teknik untuk proyek Pegasus dan Centaur. Hal ini diperlukan untuk membuat kebijakan kustom ini karena IAM tidak mendukung tag multivalued. Anda tidak dapat menandai pengguna Saanvi dengan `access-project` = `peg` dan `access-project` = `cen`. Selain itu, model AWS otorisasi tidak dapat mencocokkan kedua nilai. Untuk informasi selengkapnya, lihat Aturan untuk menandai dan IAM AWS STS. Sebagai gantinya, Anda harus menentukan secara manual dua peran yang dapat dia tanggung.

   Untuk menggunakan kebijakan ini, ganti teks placeholder yang dicetak miring dengan informasi akun Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::account-ID-without-hyphens:role/access-peg-
engineering",
```

```
                "arn:aws:iam::account-ID-without-hyphens:role/access-cen-
engineering"
            ]
        }
    ]
}
```

9. Gunakan prosedur di Langkah 4: Uji pembuatan rahasia dan Langkah 5: Uji menampilkan rahasia. Pada jendela peramban lain, konfirmasikan bahwa Saanvi dapat mengambil kedua peran tersebut. Periksa bahwa dia dapat membuat rahasia hanya untuk proyek, tim, dan pusat biaya miliknya, bergantung pada tanda peran. Konfirmasi juga bahwa dia dapat melihat perincian tentang rahasia apa pun yang dimiliki oleh tim teknik, termasuk rahasia yang baru saja dia buat.

Langkah 7: Uji memperbarui dan menghapus rahasia

Kebijakan `access-same-project-team` yang terlampir pada peran memungkinkan karyawan memperbarui dan menghapus rahasia yang ditandai dengan proyek, tim, dan pusat biaya mereka. Konfirmasikan bahwa izin Anda bekerja sesuai harapan dengan menguji peran Anda di Secrets Manager.

Untuk menguji pembaruan dan penghapusan rahasia dengan dan tanpa tag yang diperlukan

1. Masuk sebagai salah satu IAM pengguna berikut:
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
   - `access-Nikhil-cen-eng`
2. Beralih ke peran yang sesuai:
   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-peg-quality-assurance`
   - `access-cen-engineering`

Untuk informasi selengkapnya tentang beralih peran di AWS Management Console, lihatBeralih dari pengguna ke IAM peran (konsol).

3. Untuk setiap peran, coba perbarui deskripsi rahasia, lalu coba hapus rahasia berikut. Untuk informasi selengkapnya, lihat, lihat Memodifikasi Rahasia dan Menghapus dan Memulihkan Rahasia di Panduan Pengguna AWS Secrets Manager .

   Tabel berikut menunjukkan perilaku memperbarui dan menghapus ABAC rahasia untuk setiap peran.

Nama peran	Nama rahasia	Perilaku yang diharapkan
access-peg-engineering	test-access-peg-eng	Diizinkan
	test-access-uni-eng	Ditolak
	test-access-uni-qas	Ditolak
access-peg-quality-assurance	test-access-peg-qas	Diizinkan
	test-access-uni-eng	Ditolak
access-uni-engineering	test-access-uni-eng	Diizinkan
	test-access-uni-qas	Ditolak
access-peg-quality-assurance	test-access-uni-qas	Diizinkan

Ringkasan

Anda sekarang telah berhasil menyelesaikan semua langkah yang diperlukan untuk menggunakan tag untuk kontrol akses berbasis atribut ()ABAC. Anda telah mempelajari cara mendefinisikan strategi penandaan. Anda menerapkan strategi tersebut ke penanggung jawab dan sumber daya Anda. Anda membuat dan menerapkan kebijakan yang menegakkan strategi untuk Secrets Manager. Anda juga belajar bahwa ABAC skala dengan mudah ketika Anda menambahkan proyek baru dan anggota tim. Akibatnya, Anda dapat masuk ke IAM konsol dengan peran pengujian dan mengalami cara menggunakan tag untuk ABAC masuk AWS.

> **Note**
>
> Anda menambahkan kebijakan yang memungkinkan tindakan hanya berdasarkan kondisi tertentu. Jika Anda menerapkan kebijakan yang berbeda kepada pengguna atau peran Anda yang memiliki izin lebih luas, maka tindakan tersebut mungkin tidak akan dibatasi untuk memerlukan penandaan. Misalnya, jika Anda memberi pengguna izin administratif penuh menggunakan kebijakan `AdministratorAccess` AWS terkelola, kebijakan ini tidak membatasi akses tersebut. Untuk informasi selengkapnya tentang bagaimana izin ditentukan saat beberapa kebijakan dilibatkan, lihat Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun..

Sumber daya terkait

Untuk informasi terkait, lihat sumber daya berikut:

- Tentukan izin berdasarkan atribut dengan otorisasi ABAC
- AWS kunci konteks kondisi global
- Membuat peran untuk mendelegasikan izin ke pengguna IAM
- Tag untuk AWS Identity and Access Management sumber daya
- Mengontrol akses ke AWS sumber daya menggunakan tag
- Beralih dari pengguna ke IAM peran (konsol)
- IAMtutorial: Gunakan tag SAML sesi untuk ABAC

Untuk mempelajari cara memantau tag di akun Anda, lihat Memantau perubahan tag pada AWS sumber daya dengan alur kerja tanpa server dan Acara Amazon. CloudWatch

IAMtutorial: Gunakan tag SAML sesi untuk ABAC

Attribute-based access control (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke IAM sumber daya, termasuk IAM entitas (pengguna atau peran), dan ke AWS sumber daya. Ketika entitas digunakan untuk membuat permintaan AWS, mereka menjadi prinsipal dan prinsipal tersebut menyertakan tag.

Anda juga dapat meneruskan tanda sesi saat Anda mengasumsikan peran atau menggabungkan pengguna. Anda dapat menentukan kebijakan yang menggunakan kunci tanda kondisi untuk

memberikan izin kepada prinsipal Anda berdasarkan tanda mereka. Saat Anda menggunakan tanda untuk mengontrol akses ke sumber daya AWS , Anda memungkinkan tim dan sumber daya Anda tumbuh dengan lebih sedikit perubahan ke kebijakan AWS . ABACKebijakan lebih fleksibel daripada AWS kebijakan tradisional, yang mengharuskan Anda untuk mencantumkan setiap sumber daya individu. Untuk informasi lebih lanjut tentang ABAC dan keunggulannya dibandingkan kebijakan tradisional, lihatTentukan izin berdasarkan atribut dengan otorisasi ABAC.

Jika perusahaan Anda menggunakan penyedia identitas SAML berbasis (iDP) untuk mengelola identitas pengguna perusahaan, Anda dapat menggunakan SAML atribut untuk kontrol akses berbutir halus di. AWS Atribut dapat mencakup pengidentifikasi pusat biaya, alamat surel pengguna, klasifikasi departemen, dan penugasan proyek. Saat Anda meneruskan atribut tersebut sebagai tanda sesi, Anda dapat mengontrol akses ke AWS berdasarkan tanda sesi ini.

Untuk menyelesaikan ABACtutorial dengan meneruskan SAML atribut ke kepala sesi Anda, selesaikan tugas diIAMtutorial: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag, dengan perubahan yang disertakan dalam topik ini.

Prasyarat

Untuk melakukan langkah-langkah untuk menggunakan tag SAML sesiABAC, Anda harus sudah memiliki yang berikut:

- Akses ke iDP SAML berbasis tempat Anda dapat membuat pengguna pengujian dengan atribut tertentu.
- Kemampuan untuk masuk sebagai pengguna dengan izin administratif.
- Pengalaman membuat dan mengedit IAM pengguna, peran, dan kebijakan di AWS Management Console. Namun, jika Anda memerlukan bantuan mengingat proses IAM manajemen, ABAC tutorial menyediakan tautan di mana Anda dapat melihat step-by-step instruksi.
- Pengalaman menyiapkan iDP SAML berbasis di. IAM Untuk melihat detail selengkapnya dan tautan ke IAM dokumentasi terperinci, lihatMeneruskan tag sesi dengan menggunakan AssumeRoleWithSAML.

Langkah 1: Buat pengguna uji

Lewati petunjuk di Langkah 1: Buat pengguna uji. Karena identitas Anda ditentukan dalam penyedia Anda, Anda tidak perlu menambahkan IAM pengguna untuk karyawan Anda.

Langkah 2: Buat ABAC kebijakan

Ikuti petunjuk Langkah 2: Buat ABAC kebijakan untuk membuat kebijakan terkelola yang ditentukan diIAM.

Langkah 3: Buat dan konfigurasikan SAML peran

Ketika Anda menggunakan ABAC tutorial untukSAML, Anda harus melakukan langkah-langkah tambahan untuk membuat peran, mengkonfigurasi SAML iDP, dan mengaktifkan AWS Management Console akses. Untuk informasi selengkapnya, lihat Langkah 3: Buat peran.

Langkah 3A: Buat peran SAML

Buat peran tunggal yang mempercayai penyedia SAML identitas Anda dan `test-session-tags` pengguna yang Anda buat di langkah 1. ABACTutorial menggunakan peran terpisah dengan tag peran yang berbeda. Karena Anda meneruskan tag sesi dari SAML IDP Anda, Anda hanya perlu satu peran. Untuk mempelajari cara membuat peran SAML berbasis, lihatBuat peran untuk federasi SAML 2.0 (konsol).

Beri nama peran `access-session-tags`. Lampirkan kebijakan izin `access-same-project-team` ke peran tersebut. Edit kebijakan kepercayaan peran untuk menggunakan kebijakan berikut. Untuk petunjuk terperinci tentang cara mengedit hubungan kepercayaan dari suatu peran, lihat Memperbarui kebijakan kepercayaan peran .

Kebijakan kepercayaan peran berikut memungkinkan penyedia SAML identitas Anda dan `test-session-tags` pengguna untuk mengambil peran tersebut. Saat mereka mengambil peran tersebut, mereka harus meneruskan tiga tanda sesi khusus. Diperlukan tindakan `sts:TagSession` untuk memungkinkan penerusan tanda sesi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSamlIdentityAssumeRole",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:TagSession"
            ],
            "Principal": {"Federated":"arn:aws:iam::123456789012:saml-
provider/ExampleCorpProvider"},
            "Condition": {
```

```
                "StringLike": {
                    "aws:RequestTag/cost-center": "*",
                    "aws:RequestTag/access-project": "*",
                    "aws:RequestTag/access-team": [
                        "eng",
                        "qas"
                    ]
                },
                "StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/saml"}
            }
        }
    ]
}
```

`AllowSamlIdentityAssumeRole`Pernyataan tersebut memungkinkan anggota tim Engineering and Quality Assurance untuk mengambil peran ini ketika mereka bergabung AWS dari Example Corporation IDP. `ExampleCorpProvider`SAMLPenyedia didefinisikan dalamIAM. Administrator telah menyiapkan SAML pernyataan untuk meneruskan tiga tag sesi yang diperlukan. Pernyataan ini dapat meneruskan tanda tambahan, tetapi ketiga tanda ini harus ada. Atribut identitas dapat memiliki nilai apa pun untuk tanda `cost-center` dan `access-project`. Tetapi, nilai atribut `access-team` harus cocok dengan `eng` atau `qas` untuk menunjukkan bahwa identitas ada pada tim Engineering atau Jaminan Kualitas.

Langkah 3B: Konfigurasikan IDP SAML

Konfigurasikan SAML IDP Anda untuk meneruskan`cost-center`,`access-project`, dan `access-team` atribut sebagai tag sesi. Untuk informasi selengkapnya, lihat Meneruskan tag sesi dengan menggunakan AssumeRoleWithSAML.

Untuk meneruskan atribut ini sebagai tag sesi, sertakan elemen berikut dalam SAML pernyataan Anda.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:cost-center">
  <AttributeValue>987654</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-project">
  <AttributeValue>peg</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-team">
  <AttributeValue>eng</AttributeValue>
</Attribute>
```

Langkah 3C: Aktifkan akses konsol

Aktifkan akses konsol untuk SAML pengguna federasi Anda. Untuk informasi selengkapnya, lihat Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console.

Langkah 4: Uji pembuatan rahasia

Federasi ke dalam AWS Management Console menggunakan `access-session-tags` peran. Untuk informasi selengkapnya, lihat Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console. Kemudian ikuti petunjuk di Langkah 4: Uji pembuatan rahasia untuk membuat rahasia. Gunakan SAML identitas yang berbeda dengan atribut untuk mencocokkan tag yang ditunjukkan dalam ABAC tutorial. Untuk informasi selengkapnya, lihat Langkah 4: Uji pembuatan rahasia.

Langkah 5: Uji menampilkan rahasia

Ikuti petunjuk di Langkah 5: Uji menampilkan rahasia untuk melihat rahasia yang Anda buat di langkah sebelumnya. Gunakan SAML identitas yang berbeda dengan atribut untuk mencocokkan tag yang ditunjukkan dalam ABAC tutorial.

Langkah 6: Uji skalabilitas

Ikuti petunjuk di Langkah 6: Uji skalabilitas untuk menguji skalabilitas. Lakukan ini dengan menambahkan identitas baru di iDP SAML berbasis Anda dengan atribut berikut:

- `cost-center = 101010`
- `access-project = cen`
- `access-team = eng`

Langkah 7: Uji memperbarui dan menghapus rahasia

Ikuti petunjuk di Langkah 7: Uji memperbarui dan menghapus rahasia untuk memperbarui dan menghapus rahasia. Gunakan SAML identitas yang berbeda dengan atribut untuk mencocokkan tag yang ditunjukkan dalam ABAC tutorial.

> **Important**
>
> Hapus semua rahasia yang Anda buat untuk menghindari biaya tagihan. Untuk detail tentang harga di Secrets Manager, lihat AWS Secrets Manager Harga.

Ringkasan

Anda sekarang telah berhasil menyelesaikan semua langkah yang diperlukan untuk menggunakan tag SAML sesi dan tag sumber daya untuk manajemen izin.

> **Note**
>
> Anda menambahkan kebijakan yang memungkinkan tindakan hanya berdasarkan kondisi tertentu. Jika Anda menerapkan kebijakan yang berbeda kepada pengguna atau peran Anda yang memiliki izin lebih luas, maka tindakan tersebut mungkin tidak akan dibatasi untuk memerlukan penandaan. Misalnya, jika Anda memberi pengguna izin administratif penuh menggunakan kebijakan `AdministratorAccess` AWS terkelola, kebijakan ini tidak membatasi akses tersebut. Untuk informasi selengkapnya tentang bagaimana izin ditentukan saat beberapa kebijakan dilibatkan, lihat Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun..

IAMtutorial: Izinkan pengguna untuk mengelola kredensi dan pengaturan mereka MFA

Anda dapat mengizinkan pengguna untuk mengelola perangkat dan kredensialnya sendiri multi-faktor autentikasi (MFA) di halaman Kredensial keamanan. Anda dapat menggunakan AWS Management Console untuk mengonfigurasi kredensi (kunci akses, kata sandi, sertifikat penandatanganan, dan kunci SSH publik), menghapus atau menonaktifkan kredenal yang tidak diperlukan, dan mengaktifkan MFA perangkat untuk pengguna Anda. Ini berguna untuk sejumlah kecil pengguna, tetapi tugas itu dapat dengan cepat memakan waktu seiring bertambahnya jumlah pengguna. Tutorial ini menunjukkan cara mengaktifkan praktik terbaik tersebut tanpa membebani administrator Anda.

Tutorial ini menunjukkan cara mengizinkan pengguna mengakses AWS layanan, tetapi hanya ketika mereka masukMFA. Jika mereka tidak masuk dengan MFA perangkat, maka pengguna tidak dapat mengakses layanan lain.

Alur kerja ini memiliki tiga langkah dasar.

Langkah 1: Buat kebijakan untuk menegakkan login MFA

Buat kebijakan yang dikelola pelanggan yang melarang semua tindakan kecuali beberapa IAM tindakan. Pengecualian ini memungkinkan pengguna untuk mengubah kredensialnya sendiri dan mengelola MFA perangkat mereka di halaman Kredensi Keamanan. Untuk informasi

selengkapnya tentang mengakses halaman tersebut, lihat Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol).

Langkah 2: Lampirkan kebijakan ke grup pengguna uji Anda

Buat grup pengguna yang anggotanya memiliki akses penuh ke semua EC2 tindakan Amazon jika mereka masukMFA. Untuk membuat grup pengguna seperti itu, Anda melampirkan kebijakan AWS terkelola yang disebut `AmazonEC2FullAccess` dan kebijakan terkelola pelanggan yang Anda buat pada langkah pertama.

Langkah 3: Uji akses pengguna Anda

Masuk sebagai pengguna uji untuk memverifikasi bahwa akses ke Amazon EC2 diblokir hingga pengguna membuat MFA perangkat. Pengguna kemudian dapat masuk menggunakan perangkat tersebut.

Prasyarat

Untuk melakukan langkah-langkah di tutorial ini, Anda harus sudah memiliki hal-hal berikut:

- Sebuah Akun AWS yang dapat Anda masuk sebagai IAM pengguna dengan izin administratif.
- Nomor ID akun Anda, yang Anda ketikkan ke dalam kebijakan di Langkah 1.

  Untuk menemukan nomor ID akun Anda, di bilah navigasi di bagian atas halaman, pilih Dukungan lalu pilih Pusat Dukungan. Anda dapat menemukan ID akun Anda di menu Dukungan di halaman ini.
- Perangkat virtual (berbasis perangkat lunak), kunci FIDO keamanan, atau MFA perangkat berbasis perangkat keras. MFA
- IAMPengguna uji yang merupakan anggota grup pengguna sebagai berikut:

Nama pengguna	Instruksi nama pengguna	Nama grup pengguna	Tambahkan pengguna sebagai anggota	Instruksi grup pengguna
MFAUser	Pilih hanya opsi untuk Aktifkan akses konsol —	EC2MFA	MFAUser	NOTLampirkan kebijakan apa pun atau berikan izin ke grup pengguna ini.

Nama pengguna	Instruksi nama pengguna	Nama grup pengguna	Tambahkan pengguna sebagai anggota	Instruksi grup pengguna
	opsional, dan tetapkan kata sandi.			

Langkah 1: Buat kebijakan untuk menegakkan login MFA

Anda mulai dengan membuat kebijakan terkelola IAM pelanggan yang menolak semua izin kecuali yang diperlukan bagi IAM pengguna untuk mengelola kredensi dan perangkat mereka sendiri. MFA

1. Masuk ke Konsol AWS Manajemen sebagai pengguna dengan kredensi administrator. Untuk mematuhi praktik IAM terbaik, jangan masuk dengan Pengguna root akun AWS kredensi Anda.

   > **Important**
   >
   > IAMPraktik terbaik merekomendasikan bahwa Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara alih-alih menggunakan IAM pengguna dengan kredensi jangka panjang. Kami menyarankan Anda hanya menggunakan IAM pengguna untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.

2. Buka IAM konsol di https://console.aws.amazon.com/iam/.

3. Di panel navigasi, pilih Kebijakan dan kemudian pilih Buat kebijakan.

4. Pilih JSONtab dan salin teks dari dokumen JSON kebijakan berikut:AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.

5. Tempelkan teks kebijakan ke dalam kotak JSONteks. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

> **Note**
>
> Anda dapat beralih antara editor Visual dan JSONopsi kapan saja. Namun, kebijakan di atas mencakup unsur `NotAction` yang tidak didukung di editor visual. Untuk kebijakan ini, Anda akan melihat pemberitahuan di tab Editor visual. Kembali ke JSONuntuk terus bekerja dengan kebijakan ini.
> Kebijakan contoh ini tidak mengizinkan pengguna untuk mengatur ulang kata sandi saat masuk AWS Management Console untuk pertama kalinya. Kami menyarankan Anda untuk tidak memberikan izin kepada pengguna baru sampai setelah mereka masuk dan mengatur ulang kata sandi mereka.

6. Pada halaman Tinjau dan buat, ketik **`Force_MFA`** nama kebijakan. Untuk deskripsi kebijakan, ketik **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`** Di area Tag, Anda dapat menambahkan pasangan nilai kunci tag secara opsional ke kebijakan yang dikelola pelanggan. Tinjau izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul di daftar kebijakan terkelola dan siap dilampirkan.

Langkah 2: Lampirkan kebijakan ke grup pengguna uji Anda

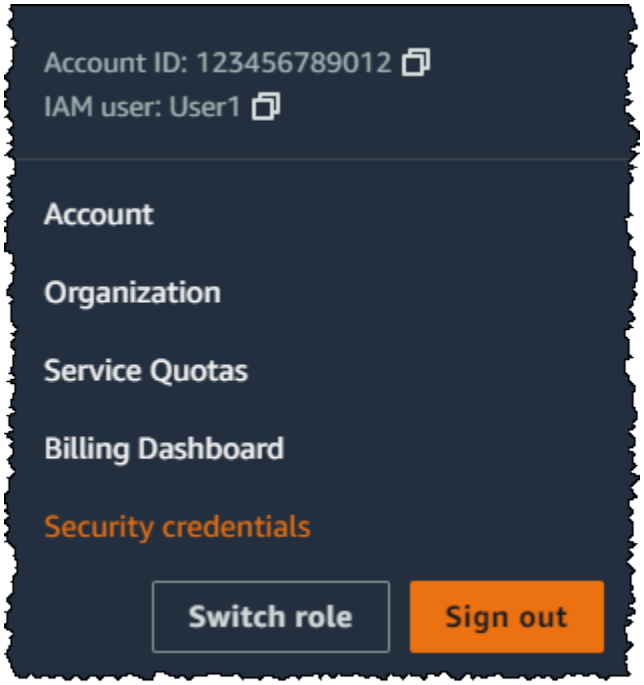
Selanjutnya Anda melampirkan dua kebijakan ke grup IAM pengguna pengujian, yang akan digunakan untuk memberikan izin MFA -protected.

1. Di panel navigasi, pilih User groups (Grup pengguna).
2. Pada kotak pencarian, ketik **`EC2MFA`**, lalu pilih nama grup (bukan kotak centang) dalam daftar.
3. Pilih tab Izin, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.
4. Pada halaman Lampirkan kebijakan izin ke EC2MFA grup, di kotak pencarian, ketik**`EC2Full`**. Kemudian pilih kotak centang di sebelah Amazon EC2FullAccess dalam daftar. Jangan menyimpan perubahan Anda.
5. Di kotak pencarian, ketik**`Force`**, lalu pilih kotak centang di sebelah Paksa_ MFA dalam daftar.
6. Pilih Lampirkan kebijakan.

Langkah 3: Uji akses pengguna Anda

Di bagian tutorial ini, Anda masuk sebagai pengguna uji dan memverifikasi bahwa kebijakan berjalan sebagaimana mestinya.

1. Masuk ke Anda Akun AWS sebagai **`MFAUser`** dengan kata sandi yang Anda tetapkan di bagian sebelumnya. GunakanURL: `https://<alias or account ID number>.signin.aws.amazon.com/console`
2. Pilih EC2untuk membuka EC2 konsol Amazon dan verifikasi bahwa pengguna tidak memiliki izin untuk melakukan apa pun.
3. Di bilah navigasi di kanan atas, pilih nama `MFAUser` pengguna, lalu pilih Kredensi keamanan.



4. Sekarang tambahkan MFA perangkat. Di bagian Otentikasi Multi-faktor (MFA), pilih MFATetapkan perangkat.

> **Note**
>
> Anda mungkin menerima kesalahan bahwa Anda tidak berwenang untuk melakukan `iam:DeleteVirtualMFADevice`. Ini bisa terjadi jika seseorang sebelumnya mulai menetapkan MFA perangkat virtual ke pengguna ini dan membatalkan prosesnya. Untuk melanjutkan, Anda atau administrator lain harus menghapus MFA perangkat

virtual pengguna yang belum ditetapkan. Untuk informasi selengkapnya, lihat Saya tidak berwenang untuk melakukan: iam: DeleteVirtual MFADevice.

5. Untuk tutorial ini, kami menggunakan perangkat virtual (berbasis MFA perangkat lunak), seperti aplikasi Google Authenticator di ponsel. Pilih aplikasi Authenticator, lalu klik Berikutnya.

   IAMmenghasilkan dan menampilkan informasi konfigurasi untuk MFA perangkat virtual, termasuk grafik kode QR. Grafik adalah representasi kunci konfigurasi rahasia yang tersedia untuk entri manual pada perangkat yang tidak mendukung kode QR.

6. Buka MFA aplikasi virtual Anda. (Untuk daftar aplikasi yang dapat Anda gunakan untuk hosting MFA perangkat virtual, lihat MFAAplikasi Virtual.) Jika MFA aplikasi virtual mendukung beberapa akun (beberapa MFA perangkat virtual), pilih opsi untuk membuat akun baru (MFAperangkat virtual baru).

7. Tentukan apakah MFA aplikasi mendukung kode QR, lalu lakukan salah satu hal berikut:

   - Dari wizard, pilih Tampilkan kode QR. Lalu gunakan aplikasi untuk memindai kode QR. Misalnya, Anda dapat memilih ikon kamera atau memilih opsi yang mirip dengan Pindai kode, lalu gunakan kamera perangkat untuk memindai kode.
   - Di wizard Siapkan perangkat, pilih Tampilkan kunci rahasia, lalu ketik kunci rahasia ke dalam MFA aplikasi Anda.

   Setelah selesai, MFA perangkat virtual mulai menghasilkan kata sandi satu kali.

8. Di wizard Menyiapkan perangkat, di Masukkan kode dari aplikasi autentikator Anda. kotak, ketik kata sandi satu kali yang saat ini muncul di MFA perangkat virtual. PilihPendaftaranMFA.

   > **Important**
   >
   > Kirim permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat berhasil dikaitkan dengan pengguna. Namun, MFA perangkat tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

   MFAPerangkat virtual sekarang siap digunakan AWS.

9. Keluar dari konsol lalu masuk sebagai **`MFAUser`** lagi. Kali ini AWS meminta Anda untuk MFA kode dari ponsel Anda. Saat Anda mendapatkannya, ketik kode di kotak, lalu pilih Kirim.
10. Pilih EC2untuk membuka EC2 konsol Amazon lagi. Perhatikan bahwa saat ini Anda dapat melihat semua informasi dan melakukan tindakan apa pun yang Anda inginkan. Jika Anda mengunjungi konsol lain sebagai pengguna ini, Anda melihat akses pesan yang ditolak. Alasannya adalah bahwa kebijakan dalam tutorial ini hanya memberikan akses ke AmazonEC2.

Sumber daya terkait

Untuk informasi tambahan, lihat topik berikut:

- AWS Otentikasi multi-faktor di IAM
- MFAmengaktifkan login

IAMIdentitas

IAMIdentitas mewakili pengguna manusia atau beban kerja terprogram yang dapat diautentikasi dan kemudian diberi wewenang untuk melakukan tindakan di. Akun AWS Identitas dapat dikaitkan dengan satu atau lebih kebijakan, yang menentukan tindakan apa yang diizinkan untuk dilakukan identitas, pada AWS sumber daya mana, dan dalam kondisi apa. IAMidentitas termasuk, IAM pengguna, IAM grup, dan IAM peran.

Anda dapat menggabungkan identitas yang ada dari penyedia identitas eksternal. Identitas ini akan mengambil IAM peran untuk mengakses AWS sumber daya. Untuk informasi selengkapnya, lihat the section called “Penyedia dan federasi identitas”.

Anda juga dapat menggunakan AWS IAM Identity Center untuk membuat dan mengelola identitas dan akses ke AWS sumber daya. IAMSet izin Pusat Identitas secara otomatis membuat IAM peran yang diperlukan untuk menyediakan akses ke sumber daya. Untuk informasi lebih lanjut, lihat Apa itu Pusat IAM Identitas?

Pengguna root akun AWS Itu adalah Akun AWS prinsip yang dibuat ketika Anda Akun AWS didirikan. Pengguna root memiliki akses ke semua AWS layanan dan sumber daya di akun. Untuk informasi selengkapnya, lihat the section called “Pengguna root IAM”.

> **Note**
>
> - Ikuti praktik terbaik Keamanan IAM saat bekerja dengan IAM identitas.
> - Ikuti praktik terbaik pengguna root untuk Anda Akun AWS saat bekerja dengan pengguna root.
> - Jika Anda mengalami masalah saat masuk, lihat Masuk ke AWS Management Console.

Pengguna root IAM

Saat pertama kali membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root. Untuk informasi selengkapnya, lihat pengguna root AWS akun.

IAMpengguna

IAMPengguna adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Untuk informasi selengkapnya, lihat IAMpengguna.

Grup pengguna IAM

Grup IAM pengguna adalah identitas yang menentukan kumpulan IAM pengguna. Untuk informasi selengkapnya, lihat Grup pengguna.

IAMperan

IAMPeran adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Ini mirip dengan IAM pengguna, tetapi tidak terkait dengan orang tertentu. Untuk informasi selengkapnya, lihat IAMperan.

Pengguna root akun AWS

Saat pertama kali membuat akun Amazon Web Services (AWS), alamat email dan kata sandi yang Anda berikan adalah kredensyal untuk pengguna root Anda, yang memiliki akses ke semua AWS layanan dan sumber daya di akun tersebut.

- Gunakan pengguna root hanya untuk melakukan tugas-tugas yang memerlukan izin tingkat root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihatTugas yang memerlukan kredensial pengguna root.
- Ikuti praktik terbaik pengguna root untuk Anda Akun AWS.
- Jika Anda mengalami masalah saat masuk, lihat Masuk ke AWS Management Console.

Untuk bantuan terkait masalah pengguna root, lihatMemecahkan masalah dengan pengguna root.

Topik berikut mendetail tugas manajemen yang terkait dengan pengguna root.

Tugas

- Otentikasi multi-faktor untuk Pengguna root akun AWS
- Ubah kata sandi untuk Pengguna root akun AWS
- Setel ulang kata sandi pengguna root yang hilang atau terlupakan

- Buat kunci akses untuk pengguna root
- Hapus kunci akses untuk pengguna root
- Tugas yang memerlukan kredensial pengguna root
- Informasi terkait

Otentikasi multi-faktor untuk Pengguna root akun AWS

Otentikasi multi-faktor (MFA) adalah mekanisme sederhana dan efektif untuk meningkatkan keamanan Anda. Faktor pertama — kata sandi Anda — adalah rahasia yang Anda hafal, juga dikenal sebagai faktor pengetahuan. Faktor lain dapat berupa faktor kepemilikan (sesuatu yang Anda miliki, seperti kunci keamanan) atau faktor warisan (sesuatu yang Anda miliki, seperti pemindaian biometrik). Untuk meningkatkan keamanan, kami sangat menyarankan Anda mengonfigurasi otentikasi multi-faktor (MFA) untuk membantu melindungi sumber daya Anda AWS .

Anda dapat mengaktifkan MFA untuk Pengguna root akun AWS dan IAM pengguna. Saat Anda mengaktifkan MFA untuk pengguna root, itu hanya memengaruhi kredensi pengguna root. Untuk informasi selengkapnya tentang cara mengaktifkan MFA IAM pengguna Anda, lihatAWS Otentikasi multi-faktor di IAM.

Sebelum Anda mengaktifkan MFA untuk pengguna root Anda, tinjau dan perbarui pengaturan akun dan informasi kontak Anda untuk memastikan bahwa Anda memiliki akses ke email dan nomor telepon. Jika MFA perangkat Anda hilang, dicuri, atau tidak berfungsi, Anda masih dapat masuk sebagai pengguna root dengan memverifikasi identitas Anda menggunakan email dan nomor telepon tersebut. Untuk mempelajari tentang proses masuk menggunakan faktor autentikasi alternatif, lihat Memulihkan identitas yang MFA dilindungi di IAM. Untuk menonaktifkan fitur ini, hubungi AWS Support.

AWS mendukung MFA jenis berikut untuk pengguna root Anda:

- Kunci sandi dan kunci keamanan
- Aplikasi otentikator virtual
- TOTPToken perangkat keras

Kunci sandi dan kunci keamanan

AWS Identity and Access Management mendukung kunci sandi dan kunci keamanan untukMFA. Berdasarkan FIDO standar, kunci sandi menggunakan kriptografi kunci publik untuk memberikan

otentikasi yang kuat dan tahan phishing yang lebih aman daripada kata sandi. AWS mendukung dua jenis kunci sandi: kunci sandi terikat perangkat (kunci keamanan) dan kunci sandi yang disinkronkan.

- Kunci keamanan: Ini adalah perangkat fisik, seperti YubiKey, digunakan sebagai faktor kedua untuk otentikasi. Satu kunci keamanan dapat mendukung beberapa akun pengguna root dan IAM pengguna.
- Kunci sandi yang disinkronkan: Ini menggunakan pengelola kredensi dari penyedia seperti Google, Apple, akun Microsoft, dan layanan pihak ketiga seperti 1Password, Dashlane, dan Bitwarden sebagai faktor kedua.

Anda dapat menggunakan autentikator biometrik bawaan, seperti Touch ID di Apple MacBooks, untuk membuka kunci pengelola kredensi dan masuk. AWS Kunci sandi dibuat dengan penyedia pilihan Anda menggunakan sidik jari, wajah, atau perangkat PIN Anda. Anda dapat menyinkronkan kunci sandi di seluruh perangkat Anda untuk memfasilitasi masuk dengan AWS, meningkatkan kegunaan dan pemulihan.

IAMtidak mendukung pendaftaran passkey lokal untuk Windows Hello. Untuk membuat dan menggunakan kunci sandi, pengguna Windows harus menggunakan otentikasi lintas perangkat di mana Anda menggunakan kunci sandi dari satu perangkat seperti perangkat seluler atau kunci keamanan perangkat keras untuk masuk di perangkat lain seperti laptop. FIDOAliansi menyimpan daftar semua produk FIDO Bersertifikat yang kompatibel dengan FIDO spesifikasi. Untuk informasi selengkapnya tentang mengaktifkan kunci sandi dan kunci keamanan, lihat. Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)

Aplikasi otentikator virtual

Aplikasi otentikator virtual berjalan di telepon atau perangkat lain dan mengemulasi perangkat fisik. Aplikasi otentikator virtual mengimplementasikan algoritma kata sandi satu kali (TOTP) berbasis waktu dan mendukung beberapa token pada satu perangkat. Pengguna harus mengetikkan kode yang valid dari perangkat saat diminta saat masuk. Setiap token yang ditetapkan untuk pengguna harus unik. Pengguna tidak dapat mengetik kode dari token pengguna lain untuk mengautentikasi.

Kami menyarankan Anda menggunakan MFA perangkat virtual sambil menunggu persetujuan pembelian perangkat keras atau saat Anda menunggu perangkat keras Anda tiba. Untuk daftar beberapa aplikasi yang didukung yang dapat Anda gunakan sebagai MFA perangkat virtual, lihat Autentikasi Multi-Faktor () MFA. Untuk petunjuk tentang pengaturan MFA perangkat virtual dengan AWS, lihatAktifkan MFA perangkat virtual untuk pengguna root (konsol).

TOTPToken perangkat keras

Perangkat keras menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali () berbasis waktu. TOTP Pengguna harus mengetik kode yang valid dari perangkat di halaman web kedua saat masuk. Setiap MFA perangkat yang ditetapkan untuk pengguna harus unik. Pengguna tidak dapat mengetik kode dari perangkat pengguna lain untuk diautentikasi. Untuk informasi tentang MFA perangkat keras yang didukung, lihat Otentikasi Multi-Faktor () MFA. Untuk petunjuk tentang cara menyiapkan TOTP token perangkat keras AWS, lihatAktifkan TOTP token perangkat keras untuk pengguna root (konsol).

Jika Anda ingin menggunakan MFA perangkat fisik, kami sarankan Anda menggunakan kunci FIDO keamanan sebagai alternatif TOTP perangkat keras. FIDOKunci keamanan menawarkan manfaat tanpa persyaratan baterai, ketahanan phishing, dan mereka mendukung banyak root dan IAM pengguna pada satu perangkat untuk meningkatkan keamanan.

Topik

- Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)
- Aktifkan MFA perangkat virtual untuk pengguna root (konsol)
- Aktifkan TOTP token perangkat keras untuk pengguna root (konsol)

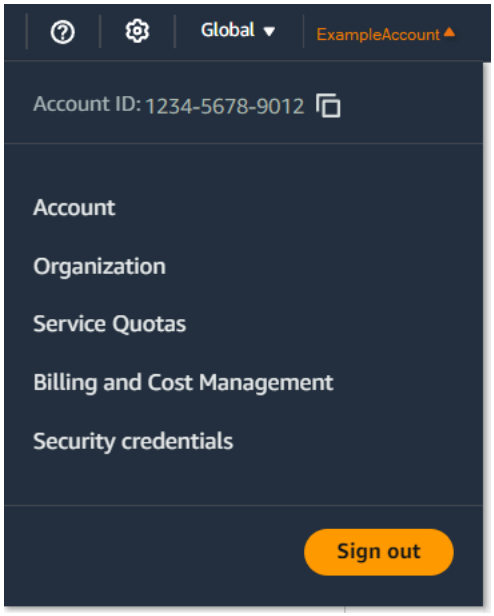
Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)

Anda dapat mengonfigurasi dan mengaktifkan kunci sandi untuk pengguna root Anda dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API.

Untuk mengaktifkan kunci sandi atau kunci keamanan untuk pengguna root Anda (konsol)

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi keamanan.



3. Pada pengguna root Anda Halaman kredensial keamanan saya, di bawah Autentikasi multi-faktor (MFA), pilih Tetapkan perangkat. MFA
4. Pada halaman nama MFA perangkat, masukkan nama Perangkat, pilih Kunci Sandi atau Kunci Keamanan, lalu pilih Berikutnya.
5. Pada Siapkan perangkat, atur kunci sandi Anda. Buat passkey dengan data biometrik seperti wajah atau sidik jari Anda, dengan pin perangkat, atau dengan memasukkan kunci FIDO keamanan ke USB port komputer Anda dan mengetuknya.
6. Ikuti petunjuk di browser Anda untuk memilih penyedia kunci sandi atau tempat Anda ingin menyimpan kunci sandi untuk digunakan di seluruh perangkat Anda.
7. Pilih Lanjutkan.

Anda sekarang telah mendaftarkan passkey Anda untuk digunakan dengan AWS. Lain kali Anda menggunakan kredensi pengguna root Anda untuk masuk, Anda harus mengautentikasi dengan kunci sandi Anda untuk menyelesaikan proses masuk.

Untuk membantu mengatasi masalah dengan kunci FIDO keamanan Anda, lihat. Memecahkan masalah kunci keamanan FIDO

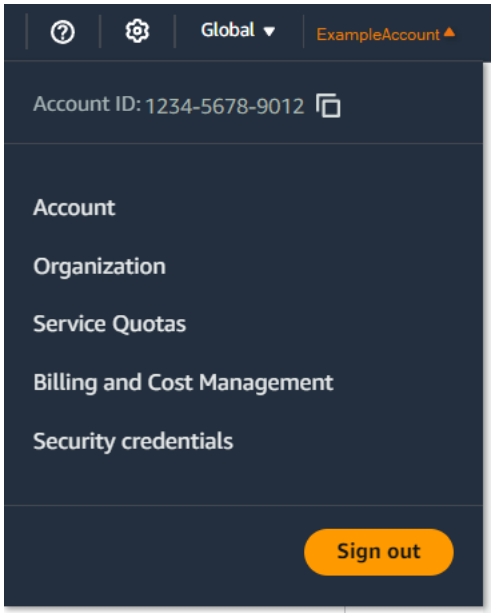
Aktifkan MFA perangkat virtual untuk pengguna root (konsol)

Anda dapat menggunakan AWS Management Console untuk mengkonfigurasi dan mengaktifkan MFA perangkat virtual untuk pengguna root Anda. Untuk mengaktifkan MFA perangkat Akun AWS, Anda harus masuk AWS menggunakan kredensi pengguna root Anda.

Untuk mengkonfigurasi dan mengaktifkan MFA perangkat virtual untuk digunakan dengan pengguna root Anda (konsol)

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sisi kanan bilah navigasi, pilih nama akun Anda, dan pilih Kredensi keamanan.



3. Di bagian Multi-Factor Authentication (MFA), pilih MFATetapkan perangkat.
4. Di wizard, ketik nama Perangkat, pilih Aplikasi Authenticator, lalu pilih Berikutnya.

   IAMmenghasilkan dan menampilkan informasi konfigurasi untuk MFA perangkat virtual, termasuk grafik kode QR. Grafik adalah representasi kunci konfigurasi rahasia yang tersedia untuk entri manual pada perangkat yang tidak mendukung kode QR.
5. Buka MFA aplikasi virtual di perangkat.

   Jika MFA aplikasi virtual mendukung beberapa MFA perangkat atau akun virtual, pilih opsi untuk membuat MFA perangkat atau akun virtual baru.
6. Cara termudah untuk mengonfigurasi aplikasi adalah menggunakan aplikasi untuk memindai kode QR. Jika Anda tidak dapat memindai kode, Anda dapat mengetik informasi konfigurasi secara manual. Kode QR dan kunci konfigurasi rahasia yang dihasilkan oleh IAM terikat dengan Anda Akun AWS dan tidak dapat digunakan dengan akun yang berbeda. Namun, mereka dapat digunakan kembali untuk mengonfigurasi MFA perangkat baru untuk akun Anda jika Anda kehilangan akses ke MFA perangkat asli.

- Untuk menggunakan kode QR untuk mengkonfigurasi MFA perangkat virtual, dari wizard, pilih Tampilkan kode QR. Lalu, ikuti petunjuk aplikasi untuk memindai kode. Misalnya, Anda mungkin perlu memilih ikon kamera atau memilih perintah seperti Kode batang akun pemindaian, lalu gunakan kamera perangkat untuk memindai kode QR.
- Di wizard Siapkan perangkat, pilih Tampilkan kunci rahasia, lalu ketik kunci rahasia ke dalam MFA aplikasi Anda.

> **Important**
>
> Buat cadangan kode QR atau kunci konfigurasi rahasia yang aman, atau pastikan Anda mengaktifkan beberapa MFA perangkat untuk akun Anda. Anda dapat mendaftarkan hingga delapan MFA perangkat dari kombinasi MFAjenis yang saat ini didukung dengan Anda Pengguna root akun AWS dan IAM pengguna. MFAPerangkat virtual mungkin menjadi tidak tersedia, misalnya, jika Anda kehilangan ponsel cerdas tempat MFA perangkat virtual di-host. Jika itu terjadi dan Anda tidak dapat masuk ke akun Anda tanpa MFA perangkat tambahan yang dilampirkan ke pengguna atau bahkan olehMemulihkan perangkat pengguna MFA root, Anda tidak akan dapat masuk ke akun Anda dan Anda harus menghubungi layanan pelanggan untuk menghapus MFA perlindungan untuk akun tersebut.

Alat mulai menghasilkan angka enam-digit.

7. Di wizard, di kotak MFAkode 1, ketik kata sandi satu kali yang saat ini muncul di MFA perangkat virtual. Tunggu hingga 30 detik untuk membuat kata sandi sekali pakai yang baru. Kemudian ketik kata sandi satu kali kedua ke dalam kotak MFAkode 2. Pilih Tambah MFA.

> **Important**
>
> Kirim permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat berhasil dikaitkan dengan pengguna tetapi MFA perangkat tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat siap digunakan dengan AWS. Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihatMFAmengaktifkan login.

Aktifkan TOTP token perangkat keras untuk pengguna root (konsol)

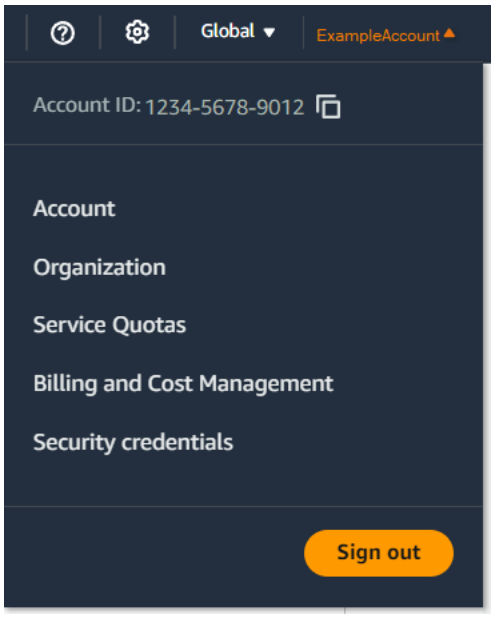
Anda dapat mengkonfigurasi dan mengaktifkan MFA perangkat fisik untuk pengguna root Anda dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API.

> **Note**
>
> Anda mungkin melihat teks yang berbeda, seperti Masuk menggunakan MFA dan Memecahkan masalah perangkat autentikasi Anda. Namun, fitur yang sama disediakan. Dalam kedua kasus tersebut, jika Anda tidak dapat memverifikasi alamat email akun dan nomor telepon Anda menggunakan faktor otentikasi alternatif, hubungi AWS Supportuntuk menonaktifkan pengaturan AndaMFA.

Untuk mengaktifkan TOTP token perangkat keras untuk pengguna root Anda (konsol)

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi keamanan.



3. Perluas bagian Autentikasi Multi-faktor (MFA).

4. Pilih Tetapkan MFA perangkat.
5. Di wizard, ketikkan nama Perangkat, pilih TOTPToken perangkat keras, lalu pilih Berikutnya.
6. Di kotak Nomor seri, ketik nomor seri yang ditemukan di bagian belakang MFA perangkat.
7. Dalam kotak MFAkode 1, ketik angka enam digit yang ditampilkan oleh perangkat. MFA Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor.

8. Tunggu 30 detik saat perangkat menyegarkan kode, lalu ketik angka enam digit berikutnya ke dalam kotak kode 2. MFA Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor kedua.
9. Pilih Tambah MFA. MFAPerangkat sekarang dikaitkan dengan Akun AWS.

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat berhasil dikaitkan dengan pengguna tetapi MFA perangkat menjadi tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Lain kali Anda menggunakan kredensi pengguna root Anda untuk masuk, Anda harus mengetikkan kode dari perangkat. MFA

Ubah kata sandi untuk Pengguna root akun AWS

Anda dapat mengubah alamat email dan kata sandi dari Kredensial Keamanan atau halaman Akun. Anda juga dapat memilih Lupa kata sandi? pada halaman AWS login untuk mengatur ulang kata sandi Anda.

Untuk mengubah kata sandi pengguna root, Anda harus masuk sebagai Pengguna root akun AWS dan bukan sebagai IAM pengguna. Untuk mempelajari cara mengatur ulang kata sandi pengguna akar yang lupa, lihat Setel ulang kata sandi pengguna root yang hilang atau terlupakan.

Untuk melindungi kata sandi Anda, penting untuk mengikuti praktik terbaik ini:

- Ubah kata sandi Anda secara berkala.
- Jaga kerahasiaan kata sandi Anda karena siapa pun yang mengetahui kata sandi Anda dapat mengakses akun Anda.
- Gunakan kata sandi yang berbeda AWS dari yang Anda gunakan di situs lain.
- Hindari kata sandi yang mudah ditebak. Ini termasuk kata sandi seperti `secret`, `password`, `amazon`, atau `123456`. Hindari juga hal-hal seperti kata-kata kamus, nama Anda, alamat email, atau informasi pribadi lainnya yang dapat diperoleh seseorang dengan mudah.

AWS Management Console

Untuk mengubah kata sandi bagi pengguna akar

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai IAM pengguna atau peran.

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda, lalu pilih Akun.
3. Pada halaman Akun, di samping Pengaturan akun, pilih Edit. Anda diminta untuk mengautentikasi ulang untuk tujuan keamanan.

   > **Note**
   >
   > Jika Anda tidak melihat opsi Edit, kemungkinan Anda tidak masuk sebagai pengguna root untuk akun Anda. Anda tidak dapat mengubah setelan akun saat masuk sebagai IAM pengguna atau peran.

4. Pada halaman Perbarui pengaturan akun, di bawah Kata Sandi, pilih Edit.
5. Pada halaman Perbarui kata sandi Anda, isi kolom untuk Kata sandi saat ini, Kata sandi baru, dan Konfirmasi kata sandi baru.

   > **Important**
   >
   > Pastikan untuk memilih kata sandi yang kuat. Meskipun Anda dapat menetapkan kebijakan kata sandi akun untuk IAM pengguna, kebijakan tersebut tidak berlaku untuk pengguna root.

   AWS mengharuskan kata sandi Anda memenuhi ketentuan berikut:

   - Itu harus memiliki minimal 8 karakter dan maksimal 128 karakter.
   - Ini harus mencakup minimal tiga dari campuran tipe karakter berikut: huruf besar, huruf kecil, angka, dan! @ # $% ^ & * () <> [] {} | _+-= simbol.
   - Itu tidak boleh identik dengan Akun AWS nama atau alamat email Anda.
6. Pilih Simpan perubahan.

AWS CLI or AWS SDK

Tugas ini tidak didukung di AWS CLI atau oleh API operasi dari salah satu AWS SDKs. Anda dapat melakukan tugas ini hanya dengan menggunakan AWS Management Console.

Setel ulang kata sandi pengguna root yang hilang atau terlupakan

Saat pertama kali membuat Akun AWS, Anda memberikan alamat email dan kata sandi. Ini adalah Pengguna root akun AWS kredensialmu. Jika Anda lupa kata sandi pengguna root Anda, Anda dapat mengatur ulang kata sandi dari file AWS Management Console.

> **Important**
>
> Mengalami masalah saat masuk AWS? Pastikan Anda berada di halaman masuk AWS yang benar untuk jenis pengguna Anda. Jika Anda adalah Pengguna root akun AWS (pemilik akun), Anda dapat masuk AWS menggunakan kredenal yang Anda atur saat membuat. Akun AWS Jika Anda adalah IAM pengguna, administrator akun Anda dapat memberi Anda kredensi yang dapat Anda gunakan untuk masuk. AWS Jika Anda perlu meminta dukungan,

jangan gunakan tautan umpan balik di halaman ini, karena formulir diterima oleh tim AWS Dokumentasi, bukan AWS Support. Sebagai gantinya, pada halaman Hubungi Kami pilih Masih tidak dapat masuk ke AWS akun Anda dan kemudian pilih salah satu opsi dukungan yang tersedia.

Untuk mengatur ulang kata sandi pengguna akar Anda:

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.

   > **Note**
   >
   > Jika Anda masuk ke AWS Management Consoledengan kredensi IAMpengguna, maka Anda harus keluar sebelum Anda dapat mengatur ulang kata sandi pengguna root. Jika Anda melihat halaman login IAM pengguna khusus akun, pilih Masuk menggunakan kredenal akun root di dekat bagian bawah halaman. Jika perlu, berikan alamat email akun Anda dan pilih Selanjutnya untuk mengakses halaman Masuk ke pengguna akar.

2. Pilih Lupa kata sandi Anda?.

   > **Note**
   >
   > Jika Anda seorang IAM pengguna, opsi ini tidak tersedia. Lupa kata sandi Anda? opsi hanya tersedia untuk akun pengguna root. IAMpengguna harus meminta administrator mereka untuk mengatur ulang kata sandi yang terlupakan. Untuk informasi selengkapnya, lihat Saya lupa kata sandi IAM pengguna untuk AWS akun saya. Jika Anda masuk melalui portal AWS akses, lihat Menyetel ulang kata sandi pengguna Pusat IAM Identitas Anda.

3. Masukkan alamat email yang terkait dengan akun tersebut. Kemudian berikan CAPTCHA teks dan pilih Lanjutkan.
4. Periksa email yang terkait dengan Anda Akun AWS untuk pesan dari Amazon Web Services. Email akan datang dari alamat yang diakhiri dengan`@verify.signin.aws`. Ikuti petunjuk di dalam email. Jika Anda tidak melihat email di akun Anda, periksa folder spam Anda. Jika Anda

tidak lagi memiliki akses ke email, lihat Saya tidak memiliki akses ke email untuk AWS akun saya di Panduan AWS Sign-In Pengguna.

Buat kunci akses untuk pengguna root

> **Warning**
>
> Kami sangat menyarankan agar Anda tidak membuat pasangan kunci akses untuk pengguna root Anda. Karena hanya beberapa tugas yang memerlukan pengguna root dan Anda biasanya jarang melakukan tugas-tugas tersebut, kami sarankan masuk ke AWS Management Console untuk melakukan tugas pengguna root. Sebelum membuat kunci akses, tinjau alternatif untuk kunci akses jangka panjang.

Meskipun kami tidak merekomendasikannya, Anda dapat membuat kunci akses untuk pengguna root Anda sehingga Anda dapat menjalankan perintah di AWS Command Line Interface (AWS CLI) atau menggunakan API operasi dari salah satu kredensi pengguna root yang AWS SDKs menggunakan. Saat Anda membuat access key, Anda membuat access key ID dan secret access key sebagai satu set. Selama pembuatan kunci akses, AWS memberi Anda satu kesempatan untuk melihat dan mengunduh bagian kunci akses rahasia dari kunci akses. Jika Anda tidak mengunduhnya atau Anda kehilangannya, Anda dapat menghapus access key itu lalu membuat yang baru. Anda dapat membuat kunci akses pengguna root dengan konsol, AWS CLI, atau AWS API.

Kunci akses yang baru dibuat memiliki status aktif, yang berarti Anda dapat menggunakan kunci akses untuk CLI dan API panggilan. Anda dapat menetapkan hingga dua kunci akses ke pengguna root.

Kunci akses yang tidak digunakan harus dinonaktifkan. Setelah kunci akses tidak aktif, Anda tidak dapat menggunakannya untuk API panggilan. Kunci tidak aktif masih dihitung terhadap batas Anda. Anda dapat membuat atau menghapus access key kapan saja. Namun, ketika Anda menghapus access key, kunci tersebut hilang selamanya dan tidak dapat diambil kembali.

AWS Management Console

Untuk membuat kunci akses untuk Pengguna root akun AWS

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai IAM pengguna atau peran.

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda, lalu pilih Kredensial Keamanan.
3. Pada bagian Access key, pilih Buat access key. Jika opsi ini tidak tersedia, maka Anda sudah memiliki jumlah maksimum kunci akses. Anda harus menghapus salah satu kunci akses yang ada sebelum Anda dapat membuat kunci baru. Untuk informasi selengkapnya, lihat Kuota IAM Objek.
4. Pada halaman Alternatives to root user access keys, tinjau rekomendasi keamanan. Untuk melanjutkan, pilih kotak centang, lalu pilih Buat kunci akses.
5. Pada halaman Retrieve access key, ID kunci Access Anda akan ditampilkan.
6. Di bawah Kunci akses rahasia, pilih Tampilkan dan kemudian salin ID kunci akses dan kunci rahasia dari jendela browser Anda dan tempelkan di tempat yang aman. Atau, Anda dapat memilih Unduh file.csv yang akan mengunduh file bernama `rootkey.csv` yang berisi ID kunci akses dan kunci rahasia. Simpan file di tempat yang aman.
7. Pilih Selesai. Ketika Anda tidak lagi memerlukan kunci akses, kami sarankan Anda menghapusnya, atau setidaknya pertimbangkan untuk menonaktifkannya sehingga tidak ada yang bisa menyalahgunakannya.

AWS CLI & SDKs

Untuk membuat kunci akses untuk pengguna root

> **Note**
>
> Untuk menjalankan perintah atau API operasi berikut sebagai pengguna root, Anda harus sudah memiliki satu active access key pair. Jika Anda tidak memiliki kunci akses, buat kunci akses pertama menggunakan tombol AWS Management Console. Kemudian, Anda dapat menggunakan kredensil dari kunci akses pertama dengan AWS CLI untuk membuat kunci akses kedua, atau untuk menghapus kunci akses.

- AWS CLI: aws iam create-access-key

  Example

```
$ aws iam create-access-key
{
    "AccessKey": {
        "UserName": "MyUserName",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2021-04-08T19:30:16+00:00"
    }
}
```

- AWS API: CreateAccessKeydi IAMAPIReferensi.

Hapus kunci akses untuk pengguna root

Anda dapat menggunakan AWS Management Console, AWS CLI atau AWS API untuk menghapus kunci akses pengguna root.

AWS Management Console

Untuk menghapus kunci akses untuk pengguna root

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai IAM pengguna atau peran.

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda, lalu pilih Kredensial Keamanan.
3. Di bagian Kunci akses, pilih tombol akses yang ingin Anda hapus, lalu, di bawah Tindakan, pilih Hapus.

   > **Note**
   >
   > Atau, Anda dapat Menonaktifkan kunci akses, alih-alih menghapusnya secara permanen. Dengan cara ini Anda dapat melanjutkan menggunakannya di masa depan tanpa harus mengubah ID kunci atau kunci rahasia. Meskipun kuncinya tidak aktif, setiap upaya untuk menggunakannya dalam permintaan AWS API gagal dengan akses kesalahan ditolak.
4. Pada <access key ID>kotak dialog Hapus, pilih Nonaktifkan, masukkan ID kunci akses untuk mengonfirmasi bahwa Anda ingin menghapusnya, lalu pilih Hapus.

AWS CLI & SDKs

Untuk menghapus kunci akses untuk pengguna root

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai IAM pengguna atau peran.

- AWS CLI: aws iam delete-access-key

  Example

  ```
  $ aws iam delete-access-key \
      --access-key-id AKIAIOSFODNN7EXAMPLE
  ```

  Perintah ini tidak menghasilkan output saat berhasil.
- AWS API: DeleteAccessKey

Tugas yang memerlukan kredensial pengguna root

Kami menyarankan Anda mengonfigurasi pengguna administratif AWS IAM Identity Center untuk melakukan tugas sehari-hari dan mengakses AWS sumber daya. Namun, Anda dapat melakukan tugas yang tercantum di bawah ini hanya saat masuk sebagai pengguna root akun.

Untuk bantuan terkait masalah pengguna root, lihatMemecahkan masalah dengan pengguna root.

Tugas Manajemen Akun

- Ubah pengaturan akun Anda. Ini mencakup nama akun, alamat email, kata sandi pengguna root, dan access key pengguna root. Pengaturan akun lainnya, seperti informasi kontak, preferensi mata uang pembayaran, dan Wilayah AWS, tidak memerlukan kredensi pengguna root.

- Kembalikan izin IAM pengguna. Jika satu-satunya IAM administrator secara tidak sengaja mencabut izin mereka sendiri, Anda dapat masuk sebagai pengguna root untuk mengedit kebijakan dan memulihkan izin tersebut.
- Tutup Anda Akun AWS.

  Untuk informasi selengkapnya, lihat topik berikut.

  - Bagaimana cara menetapkan kepemilikan saya Akun AWS ke entitas lain? .
  - Bagaimana cara menutup saya Akun AWS? .
  - Tutup yang berdiri sendiri Akun AWS.

Tugas Penagihan

- Aktifkan IAM akses ke konsol Billing and Cost Management.
- Beberapa tugas Penagihan terbatas pada pengguna root. Lihat Mengelola Panduan AWS Billing Pengguna Akun AWS di untuk informasi selengkapnya.
- Lihat faktur pajak tertentu. IAMPengguna dengan aws-portal: ViewBilling izin dapat melihat dan mengunduh VAT faktur dari AWS Eropa, tetapi tidak Inc. atau AWS Amazon Internet Services Private Limited (AISPL).

AWS GovCloud (US) Tugas

- Daftar AWS GovCloud (US).
- Minta kunci akses pengguna root AWS GovCloud (US) akun dari AWS Support.

EC2Tugas Amazon

- Mendaftar sebagai penjual di Marketplace Instans Cadangan.

AWS KMS Tugas

- Jika AWS Key Management Service kunci menjadi tidak dapat dikelola, administrator dapat memulihkannya dengan menghubungi AWS Support; Namun, AWS Support menanggapi nomor telepon utama pengguna root Anda untuk otorisasi dengan mengonfirmasi tiket. OTP

Tugas Amazon Mechanical Turk

- Tautkan Anda Akun AWS ke akun MTurk Pemohon Anda.

Tugas Layanan Penyimpanan Sederhana Amazon

- Konfigurasikan bucket Amazon S3 untuk mengaktifkan MFA (otentikasi multi-faktor).
- Edit atau hapus kebijakan bucket Amazon S3 yang menyangkal semua prinsipal.

Tugas Layanan Antrian Sederhana Amazon

- Mengedit atau menghapus kebijakan SQS sumber daya Amazon yang menyangkal semua prinsipal.

Informasi terkait

Artikel berikut memberikan informasi tambahan tentang bekerja dengan pengguna root.

- Apa saja praktik terbaik untuk mengamankan sumber daya saya Akun AWS dan sumber dayanya?
- Bagaimana cara membuat aturan EventBridge acara untuk memberi tahu saya bahwa pengguna root saya digunakan?
- Memantau dan memberi tahu aktivitas Pengguna root akun AWS
- Pantau aktivitas pengguna IAM root

IAMpengguna

> **Important**
>
> IAMPraktik terbaik merekomendasikan bahwa Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara alih-alih menggunakan IAM pengguna dengan kredensi jangka panjang. Kami menyarankan Anda hanya menggunakan IAM pengguna untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.

IAMPengguna adalah entitas yang Anda buat di Akun AWS. IAMPengguna mewakili pengguna manusia atau beban kerja yang menggunakan IAM pengguna untuk berinteraksi dengan AWS sumber daya. Seorang IAM pengguna terdiri dari nama dan kredensi.

Seorang IAM pengguna dengan izin administrator tidak sama dengan. Pengguna root akun AWS Untuk informasi lebih lanjut tentang pengguna akar, lihat Pengguna root akun AWS.

Bagaimana AWS mengidentifikasi pengguna IAM

Saat Anda membuat IAM pengguna, IAM buat cara-cara berikut untuk mengidentifikasi pengguna tersebut:

- Sebuah “nama ramah” untuk IAM pengguna, yang merupakan nama yang Anda tentukan ketika Anda membuat IAM pengguna, seperti `Richard` atau`Anaya`. Ini adalah nama yang Anda lihat di AWS Management Console.
- Nama Sumber Daya Amazon (ARN) untuk IAM pengguna. Anda menggunakan ARN ketika Anda perlu mengidentifikasi IAM pengguna secara unik di semua. AWS Misalnya, Anda dapat menggunakan ARN untuk menentukan IAM pengguna sebagai `Principal` dalam IAM kebijakan untuk bucket Amazon S3. ARNUntuk IAM pengguna mungkin terlihat seperti berikut:

  `arn:aws:iam::`*`account-ID-without-hyphens`*`:user/Richard`
- Pengenal unik untuk IAM pengguna. ID ini dikembalikan hanya ketika Anda menggunakanAPI, Alat untuk Windows PowerShell, atau AWS CLI untuk membuat IAM pengguna; Anda tidak melihat ID ini di konsol.

Untuk informasi selengkapnya tentang pengidentifikasi ini, lihat IAMpengidentifikasi.

IAMpengguna dan kredensialnya

Anda dapat mengakses dengan berbagai AWS cara tergantung pada kredensi IAM pengguna:

- Kata sandi konsol: Kata sandi yang dapat diketik IAM pengguna untuk masuk ke sesi interaktif seperti AWS Management Console. Menonaktifkan kata sandi (akses konsol) untuk IAM pengguna mencegah mereka masuk ke AWS Management Console menggunakan kredenal masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan.
- Kunci akses: Digunakan untuk melakukan panggilan terprogram ke AWS. Namun, ada alternatif yang lebih aman untuk dipertimbangkan sebelum Anda membuat kunci akses untuk IAM

pengguna. Untuk informasi selengkapnya, lihat Pertimbangan dan alternatif untuk kunci akses jangka panjang di. Referensi Umum AWS Jika IAM pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows PowerShell AWS API, atau AWS Console Mobile Application.

- SSHkunci untuk digunakan dengan CodeCommit: Kunci SSH publik dalam SSH format Terbuka yang dapat digunakan untuk mengautentikasi dengan CodeCommit.
- Sertifikat server:SSL/TLSsertifikat yang dapat Anda gunakan untuk mengautentikasi dengan beberapa AWS layanan. Kami menyarankan Anda menggunakan AWS Certificate Manager (ACM) untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Gunakan IAM hanya ketika Anda harus mendukung HTTPS koneksi di wilayah yang tidak didukung olehACM. Untuk mempelajari wilayah mana yang mendukungACM, lihat AWS Certificate Manager titik akhir dan kuota di. Referensi Umum AWS

Anda dapat memilih kredensi yang tepat untuk pengguna AndaIAM. Ketika Anda menggunakan AWS Management Console untuk membuat IAM pengguna, Anda harus memilih untuk setidaknya menyertakan kata sandi konsol atau kunci akses. Secara default, IAM pengguna baru yang dibuat menggunakan AWS CLI atau tidak AWS API memiliki kredensi apa pun. Anda harus membuat jenis kredensi untuk IAM pengguna berdasarkan kasus penggunaan Anda.

Anda memiliki opsi berikut untuk mengelola kata sandi, kunci akses, dan perangkat otentikasi multi-faktor ()MFA:

- Kelola kata sandi untuk IAM pengguna Anda. Buat dan ubah kata sandi yang mengizinkan akses ke AWS Management Console. Atur kebijakan kata sandi untuk menerapkan kerumitan kata sandi minimum. Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri.
- Kelola kunci akses untuk IAM pengguna Anda. Buat dan perbarui access key untuk akses terprogram ke sumber daya di akun Anda.
- Aktifkan otentikasi multi-faktor (MFA) untuk pengguna. IAM Sebagai praktik terbaik, kami menyarankan Anda memerlukan otentikasi multi-faktor untuk semua IAM pengguna di akun Anda. DenganMFA, IAM pengguna harus memberikan dua bentuk identifikasi: Pertama, mereka memberikan kredensi yang merupakan bagian dari identitas pengguna mereka (kata sandi atau kunci akses). Selain itu, mereka menyediakan kode numerik sementara yang dihasilkan pada perangkat keras atau oleh aplikasi pada smartphone atau tablet.
- Temukan kata sandi dan kunci akses yang tidak digunakan. Siapa pun yang memiliki kata sandi atau kunci akses untuk akun Anda atau IAM pengguna di akun Anda memiliki akses ke AWS

sumber daya Anda. Praktik keamanan terbaik adalah menghapus kata sandi dan kunci akses saat IAM pengguna tidak lagi membutuhkannya.

- Unduh laporan kredenal untuk akun Anda. Anda dapat membuat dan mengunduh laporan kredensi yang mencantumkan semua IAM pengguna di akun Anda dan status berbagai kredensialnya, termasuk kata sandi, kunci akses, dan perangkat. MFA Untuk kata sandi dan access key, laporan kredensial menunjukkan bagaimana kata sandi atau access key telah digunakan baru-baru ini.

IAMpengguna dan izin

Secara default, IAM pengguna baru tidak memiliki izin untuk melakukan apa pun. Mereka tidak berwenang untuk melakukan AWS operasi apa pun atau mengakses AWS sumber daya apa pun. Keuntungan memiliki IAM pengguna individu adalah Anda dapat menetapkan izin satu per satu untuk setiap pengguna. Anda dapat menetapkan izin administratif untuk beberapa pengguna, yang kemudian dapat mengelola AWS sumber daya Anda dan bahkan dapat membuat dan mengelola pengguna lain. IAM Namun, dalam kebanyakan kasus, Anda ingin membatasi izin pengguna hanya pada tugas (AWS tindakan atau operasi) dan sumber daya yang diperlukan untuk pekerjaan itu.

Bayangkan seorang pengguna bernama Diego. Saat Anda membuat IAM penggunaDiego, Anda membuat kata sandi untuknya dan melampirkan izin yang memungkinkannya meluncurkan EC2 instance Amazon tertentu dan membaca (GET) informasi dari tabel di RDS database Amazon. Untuk prosedur tentang cara membuat IAM pengguna dan memberi mereka kredensional dan izin awal, lihat. Buat IAM pengguna di Akun AWS Untuk prosedur tentang cara mengubah izin untuk pengguna yang sudah ada, lihat Mengubah izin untuk pengguna IAM. Untuk prosedur tentang cara mengubah kata sandi atau access key pengguna, lihat Kelola kata sandi pengguna di AWS dan Mengelola kunci akses untuk IAM pengguna.

Anda juga dapat menambahkan batas izin ke pengguna Anda. IAM Batas izin adalah fitur lanjutan yang memungkinkan Anda menggunakan kebijakan AWS terkelola untuk membatasi izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada pengguna atau peran. IAM Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

IAMpengguna dan akun

Setiap IAM pengguna dikaitkan dengan satu dan hanya satu Akun AWS. Karena IAM pengguna didefinisikan di dalam Anda Akun AWS, mereka tidak perlu memiliki metode pembayaran yang tersimpan AWS. Setiap AWS aktivitas yang dilakukan oleh IAM pengguna di akun Anda akan ditagih ke akun Anda.

Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

IAMpengguna sebagai akun layanan

IAMPengguna adalah sumber daya IAM yang memiliki kredensi dan izin terkait. IAMPengguna dapat mewakili seseorang atau aplikasi yang menggunakan kredensialnya untuk membuat AWS permintaan. Ini biasanya disebut sebagai akun layanan. Jika Anda memilih untuk menggunakan kredensi jangka panjang IAM pengguna dalam aplikasi Anda, jangan menanamkan kunci akses langsung ke kode aplikasi Anda. Itu AWS SDKs dan AWS Command Line Interface memungkinkan Anda untuk meletakkan kunci akses di lokasi yang diketahui sehingga Anda tidak harus menyimpannya dalam kode. Untuk informasi selengkapnya, lihat Mengelola Kunci Akses IAM Pengguna dengan Benar di Referensi Umum AWS. Atau, dan sebagai praktik terbaik, Anda dapat menggunakan kredenal keamanan sementara (IAMperan) alih-alih kunci akses jangka panjang.

Cara IAM pengguna masuk AWS

Untuk masuk ke IAM pengguna AWS Management Console sebagai pengguna, Anda harus memberikan ID akun atau alias akun Anda selain nama pengguna dan kata sandi Anda. Ketika administrator Anda membuat IAM pengguna Anda di konsol, mereka seharusnya telah mengirimi Anda kredensi masuk Anda, termasuk nama pengguna Anda dan halaman masuk URL ke akun Anda yang menyertakan ID akun atau alias akun Anda.

```
https://My_AWS_Account_ID.signin.aws.amazon.com/console/
```

> **Kiat**
>
> Untuk membuat bookmark untuk halaman login akun Anda di browser web Anda, Anda harus secara manual mengetikkan login URL untuk akun Anda di entri bookmark. Jangan gunakan fitur bookmark browser web Anda karena pengalihan dapat mengaburkan proses masuk. URL

Anda juga dapat masuk di titik akhir masuk umum berikut dan mengetikkan ID akun atau alias akun Anda secara manual:

```
https://console.aws.amazon.com/
```

Untuk kenyamanan, halaman AWS masuk menggunakan cookie browser untuk mengingat nama IAM pengguna dan informasi akun. Lain kali pengguna pergi ke halaman mana pun di AWS Management Console, konsol menggunakan cookie untuk mengarahkan pengguna ke halaman masuk akun.

Anda hanya memiliki akses ke AWS sumber daya yang ditentukan administrator dalam kebijakan yang dilampirkan pada identitas IAM pengguna Anda. Untuk bekerja di konsol, Anda harus memiliki izin untuk melakukan tindakan yang dilakukan konsol, seperti membuat daftar dan membuat AWS sumber daya. Untuk informasi selengkapnya, silakan lihat Manajemen akses untuk AWS sumber daya dan Contoh kebijakan berbasis identitas IAM.

> **Note**
>
> Jika organisasi Anda memiliki sistem identitas yang ada, Anda mungkin ingin membuat opsi sign-on (SSO) tunggal. SSOmemberi pengguna akses ke AWS Management Console akun Anda tanpa mengharuskan mereka memiliki identitas IAM pengguna. SSOjuga menghilangkan kebutuhan bagi pengguna untuk masuk ke situs organisasi Anda dan secara AWS terpisah. Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

Rincian login masuk masuk CloudTrail

Jika Anda mengaktifkan CloudTrail untuk mencatat peristiwa login ke log Anda, Anda harus mengetahui caranya CloudTrail memilih tempat untuk mencatat peristiwa.

- Jika pengguna Anda masuk langsung ke konsol, mereka diarahkan ke titik akhir masuk global atau regional, berdasarkan apakah konsol layanan yang dipilih mendukung wilayah. Misalnya, halaman beranda konsol utama mendukung wilayah, jadi jika Anda masuk ke yang berikutURL:

  ```
  https://alias.signin.aws.amazon.com/console
  ```

  Anda dialihkan ke titik akhir masuk regional seperti`https://us-east-2.signin.aws.amazon.com`, menghasilkan entri CloudTrail log regional di log wilayah pengguna:

  Di sisi lain, konsol Amazon S3 tidak mendukung wilayah, jadi jika Anda masuk ke yang berikut URL

  ```
  https://alias.signin.aws.amazon.com/console/s3
  ```

AWS mengarahkan Anda ke titik akhir masuk global dihttps://signin.aws.amazon.com, menghasilkan entri log global. CloudTrail

- Anda dapat meminta titik akhir login regional tertentu secara manual dengan masuk ke halaman beranda konsol utama berkemampuan wilayah menggunakan URL sintaks seperti berikut:

```
https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
```

AWS mengarahkan Anda ke titik akhir masuk ap-southeast-1 regional dan menghasilkan peristiwa log regional. CloudTrail

Untuk informasi selengkapnya tentang CloudTrail danIAM, lihat Logging IAM peristiwa dengan CloudTrail.

Jika pengguna memerlukan akses terprogram untuk bekerja dengan akun Anda, Anda dapat membuat access key pair (ID kunci akses dan kunci akses rahasia) untuk setiap pengguna. Namun, ada alternatif yang lebih aman untuk dipertimbangkan sebelum Anda membuat kunci akses untuk pengguna. Untuk informasi selengkapnya, lihat Pertimbangan dan alternatif untuk kunci akses jangka panjang di. Referensi Umum AWS

MFAmengaktifkan login

Pengguna yang dikonfigurasi dengan perangkat otentikasi multi-faktor (MFA) harus menggunakan MFA perangkat mereka untuk masuk ke perangkat. AWS Management Console Setelah pengguna memasukkan kredensialnya masuk, AWS periksa akun pengguna untuk melihat apakah MFA diperlukan untuk pengguna tersebut. Topik berikut memberikan informasi tentang cara pengguna menyelesaikan login saat MFA diperlukan.

Topik

- Beberapa MFA perangkat diaktifkan
- FIDOkunci keamanan
- MFAPerangkat virtual
- TOTPToken perangkat keras

Beberapa MFA perangkat diaktifkan

Jika pengguna masuk AWS Management Console sebagai pengguna Akun AWS root atau IAM pengguna dengan beberapa MFA perangkat diaktifkan untuk akun itu, mereka hanya perlu menggunakan satu MFA perangkat untuk masuk. Setelah pengguna mengautentikasi dengan kata sandi pengguna, mereka memilih jenis MFA perangkat yang ingin mereka gunakan untuk menyelesaikan otentikasi. Kemudian pengguna diminta untuk mengautentikasi dengan jenis perangkat yang mereka pilih.

FIDOkunci keamanan

Jika MFA diperlukan untuk pengguna, halaman login kedua akan muncul. Pengguna perlu mengetuk kunci FIDO keamanan.

> **Note**
>
> Pengguna Google Chrome tidak boleh memilih salah satu opsi yang tersedia pada pop-up yang meminta Verifikasi identitas Anda dengan amazon.com. Anda hanya perlu mengetuk tombol keamanan.

Tidak seperti MFA perangkat lain, kunci FIDO keamanan tidak keluar dari sinkron. Administrator dapat menonaktifkan kunci FIDO keamanan jika hilang atau rusak. Untuk informasi selengkapnya, lihat Menonaktifkan MFA perangkat (konsol).

Untuk informasi tentang browser yang mendukung WebAuthn dan FIDO mematuhi perangkat yang AWS mendukung, lihat. Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

MFAPerangkat virtual

Jika MFA diperlukan untuk pengguna, halaman login kedua akan muncul. Di kotak MFAkode, pengguna harus memasukkan kode numerik yang disediakan oleh MFA aplikasi.

Jika MFA kodenya benar, pengguna dapat mengakses file AWS Management Console. Jika kode salah, pengguna dapat mencoba kembali dengan kode lain.

MFAPerangkat virtual bisa keluar dari sinkron. Jika pengguna tidak dapat masuk ke AWS Management Console setelah beberapa kali mencoba, pengguna diminta untuk menyinkronkan

perangkat virtualMFA. Pengguna dapat mengikuti petunjuk di layar untuk menyinkronkan perangkat virtual. MFA Untuk informasi tentang cara menyinkronkan perangkat atas nama pengguna di perangkat Anda Akun AWS, lihatSinkronisasi ulang perangkat virtual dan perangkat keras MFA.

TOTPToken perangkat keras

Jika MFA diperlukan untuk pengguna, halaman login kedua akan muncul. Di kotak MFAkode, pengguna harus memasukkan kode numerik yang disediakan oleh TOTP token perangkat keras.

Jika MFA kodenya benar, pengguna dapat mengakses file AWS Management Console. Jika kode salah, pengguna dapat mencoba kembali dengan kode lain.

TOTPToken perangkat keras bisa tidak sinkron. Jika pengguna tidak dapat masuk AWS Management Console setelah beberapa kali mencoba, pengguna akan diminta untuk menyinkronkan perangkat MFA token. Pengguna dapat mengikuti petunjuk di layar untuk menyinkronkan perangkat token. MFA Untuk informasi tentang cara menyinkronkan perangkat atas nama pengguna di perangkat Anda Akun AWS, lihatSinkronisasi ulang perangkat virtual dan perangkat keras MFA.

Buat IAM pengguna di Akun AWS

> **Important**
>
> IAMPraktik terbaik merekomendasikan bahwa Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara alih-alih menggunakan IAM pengguna dengan kredensi jangka panjang. Kami menyarankan Anda hanya menggunakan IAM pengguna untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.

Proses menciptakan IAM pengguna dan memungkinkan pengguna untuk melakukan tugas terdiri dari langkah-langkah berikut:

1. Buat pengguna di AWS Management Console AWS CLI, Alat untuk Windows PowerShell, atau menggunakan AWS API operasi. Jika Anda membuat pengguna di AWS Management Console, maka langkah 1-4 ditangani secara otomatis, berdasarkan pilihan Anda. Jika Anda membuat IAM pengguna secara terprogram, maka Anda harus melakukan masing-masing langkah tersebut secara individual.
2. Buat kredensial untuk pengguna, tergantung pada jenis akses yang diperlukan pengguna:

- Aktifkan akses konsol — opsional: Jika pengguna perlu mengakses AWS Management Console, buat kata sandi untuk pengguna. Menonaktifkan akses konsol untuk pengguna mencegah mereka masuk ke AWS Management Console menggunakan nama pengguna dan kata sandi mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan.

> **Tip**
>
> Buat hanya kredenal yang dibutuhkan pengguna. Misalnya, untuk pengguna yang membutuhkan akses hanya melalui AWS Management Console, jangan membuat kunci akses.

3. Berikan izin pengguna untuk melakukan tugas yang diperlukan. Kami menyarankan Anda menempatkan IAM pengguna Anda dalam grup dan mengelola izin melalui kebijakan yang dilampirkan ke grup tersebut. Namun, Anda juga dapat memberikan izin dengan melampirkan kebijakan izin langsung ke pengguna. Jika Anda menggunakan konsol untuk menambahkan pengguna, Anda dapat menyalin izin dari pengguna yang ada ke pengguna baru.

   Anda juga dapat menambahkan batas izin untuk membatasi izin pengguna dengan menentukan kebijakan yang menentukan izin maksimum yang dapat dimiliki pengguna. Batas izin tidak memberikan izin apa pun.

   Untuk petunjuk cara membuat kebijakan izin khusus yang akan digunakan untuk memberikan izin atau menetapkan batas izin, lihat. Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan
4. (Opsional) Tambahkan metadata ke pengguna dengan melampirkan tag. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
5. Berikan informasi masuk yang diperlukan kepada pengguna. Ini termasuk kata sandi dan konsol URL untuk halaman masuk akun tempat pengguna memberikan kredensi tersebut. Untuk informasi selengkapnya, lihat Cara IAM pengguna masuk AWS.
6. (Opsional) Konfigurasikan otentikasi multi-faktor (MFA) untuk pengguna. MFAmengharuskan pengguna untuk memberikan one-time-use kode setiap kali dia masuk ke AWS Management Console.
7. (Opsional) Berikan izin kepada IAM pengguna untuk mengelola kredensi keamanan mereka sendiri. (Secara default, IAM pengguna tidak memiliki izin untuk mengelola kredensialnya sendiri.) Untuk informasi selengkapnya, lihat Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri.

> **Note**
>
> Jika Anda menggunakan konsol untuk membuat pengguna dan Anda memilih Pengguna harus membuat kata sandi baru saat masuk berikutnya (disarankan), pengguna memiliki izin yang diperlukan.

Untuk informasi tentang izin yang Anda butuhkan untuk membuat pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Untuk petunjuk cara membuat IAM pengguna untuk kasus penggunaan tertentu, lihat topik berikut:

- Buat IAM pengguna untuk akses darurat
- Buat IAM pengguna untuk beban kerja yang tidak dapat menggunakan peran IAM

Lihat IAM pengguna

Anda dapat mencantumkan IAM pengguna di grup Anda Akun AWS atau di IAM grup tertentu, dan mencantumkan semua IAM grup tempat pengguna berada. Untuk informasi tentang izin yang Anda butuhkan untuk membuat daftar pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Untuk daftar semua IAM pengguna di akun

- AWS Management Console: Di panel navigasi, pilih Pengguna. Konsol menampilkan IAM pengguna di file Anda Akun AWS.
- AWS CLI: aws iam list-users
- AWS API: ListUsers

Untuk mencantumkan IAM pengguna dalam grup pengguna tertentu

- AWS Management Console: Di panel navigasi, pilih Grup pengguna, pilih nama grup pengguna, lalu pilih tab Pengguna.
- AWS CLI: aws iam get-group
- AWS API: GetGroup

Untuk mencantumkan semua IAM grup tempat pengguna berada

- AWS Management Console: Di panel navigasi, pilih Pengguna, pilih nama pengguna, lalu pilih tab Grup.
- AWS CLI: aws iam list-groups-for -pengguna
- AWS API: ListGroupsForUser

Langkah selanjutnya

Setelah memiliki daftar IAM pengguna, Anda dapat mengganti nama, menghapus, atau menonaktifkan IAM pengguna menggunakan prosedur berikut.

- Ganti nama pengguna IAM
- Menghapus atau menonaktifkan pengguna IAM

Ganti nama pengguna IAM

> **Note**
>
> Sebagai praktik terbaik, kami menyarankan Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara. Jika Anda mengikuti praktik terbaik, Anda tidak mengelola IAM pengguna dan grup. Sebagai gantinya, pengguna dan grup Anda dikelola di luar AWS dan dapat mengakses AWS sumber daya sebagai identitas federasi. Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses AWS layanan dengan menggunakan kredensi yang disediakan melalui sumber identitas. Identitas federasi menggunakan grup yang ditentukan oleh penyedia identitas mereka. Jika Anda menggunakan AWS IAM Identity Center, lihat Mengelola identitas di Pusat IAM Identitas di Panduan AWS IAM Identity Center Pengguna untuk informasi tentang membuat pengguna dan grup di Pusat IAM Identitas.

Amazon Web Services menawarkan beberapa alat untuk mengelola IAM pengguna di situs Anda Akun AWS. Anda dapat mencantumkan IAM pengguna di akun Anda atau di grup pengguna, atau mencantumkan semua IAM grup yang menjadi anggota pengguna. Anda dapat mengganti nama

atau mengubah jalur IAM pengguna. Jika Anda beralih menggunakan identitas federasi alih-alih IAM pengguna, Anda dapat menghapus IAM pengguna dari AWS akun Anda, atau menonaktifkan pengguna.

Untuk informasi selengkapnya tentang menambahkan, mengubah, atau menghapus kebijakan terkelola untuk IAM pengguna, lihatMengubah izin untuk pengguna IAM. Untuk informasi tentang mengelola kebijakan sebaris bagi IAM pengguna, lihat Menambahkan dan menghapus izin identitas IAMEdit IAM kebijakan, danHapus IAM kebijakan. Sebagai praktik terbaik, gunakan kebijakan terkelola alih-alih kebijakan inline. AWS kebijakan terkelola memberikan izin untuk banyak kasus penggunaan umum. Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan oleh semua pelanggan. AWS Oleh karena itu, kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan terkelola pelanggan yang spesifik untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat AWS kebijakan terkelola. Untuk informasi selengkapnya tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

Untuk mempelajari tentang memvalidasi IAM kebijakan, lihatValidasi kebijakan IAM.

> **Tip**
>
> IAMAccess Analyzer dapat menganalisis layanan dan tindakan yang digunakan IAM peran Anda, lalu menghasilkan kebijakan berbutir halus yang dapat Anda gunakan. Setelah menguji setiap kebijakan yang dihasilkan, Anda dapat menerapkan kebijakan tersebut ke lingkungan produksi. Ini memastikan bahwa Anda hanya memberikan izin yang diperlukan untuk beban kerja Anda. Untuk informasi selengkapnya tentang pembuatan kebijakan, lihat pembuatan kebijakan IAM Access Analyzer.

Untuk informasi tentang mengelola kata sandi IAM pengguna, lihatKelola kata sandi untuk pengguna IAM.

Mengubah nama pengguna IAM

Untuk mengubah nama atau jalur pengguna, Anda harus menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API. Tidak ada opsi di konsol untuk mengubah nama pengguna. Untuk informasi tentang izin yang Anda butuhkan untuk mengubah nama pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Saat Anda mengubah nama atau alur pengguna, hal berikut terjadi:

- Kebijakan apa pun yang terlampir pada pengguna akan tetap melekat ke pengguna dengan nama baru.
- Pengguna tetap berada di IAM grup yang sama dengan nama baru.
- ID unik untuk pengguna tetap sama. Untuk informasi lebih lanjut tentang unikIDs, lihatPengidentifikasi unik.
- Semua kebijakan sumber daya atau peran yang mengacu pada pengguna sebagai utama (pengguna diberi akses) secara otomatis diperbarui untuk menggunakan nama atau jalur baru. Misalnya, kebijakan berbasis antrian apa pun di Amazon SQS atau kebijakan berbasis sumber daya di Amazon S3 diperbarui secara otomatis untuk menggunakan nama dan jalur baru.

IAMtidak secara otomatis memperbarui kebijakan yang merujuk ke pengguna sebagai sumber daya untuk menggunakan nama atau jalur baru; Anda harus melakukannya secara manual. Misalnya, bayangkan pengguna tersebut `Richard` memiliki kebijakan terlampir padanya yang memungkinkannya mengelola kredensial keamanannya. Jika administrator mengubah nama `Richard` untuk `Rich`, administrator juga perlu memperbarui kebijakan tersebut untuk mengubah sumber daya dari:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

ke ini:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

Hal ini juga berlaku jika administrator mengubah alur; administrator perlu memperbarui kebijakan agar mencerminkan jalur baru bagi pengguna.

Untuk mengganti nama pengguna

- AWS CLI: aws iam update-user
- AWS API: UpdateUser

Menghapus atau menonaktifkan pengguna IAM

Anda dapat menghapus IAM pengguna dari Anda Akun AWS jika pengguna tersebut berhenti dari perusahaan Anda. Jika pengguna pergi sementara, Anda dapat menonaktifkan akses pengguna alih-alih menghapusnya dari akun seperti yang dijelaskan dalam. Menonaktifkan pengguna IAM

Prasyarat — Lihat akses pengguna

Sebelum menghapus pengguna, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus pengguna IAM (konsol)

Ketika Anda menggunakan AWS Management Console untuk menghapus IAM pengguna, IAM secara otomatis menghapus informasi berikut untuk Anda:

- Pengguna
- Setiap keanggotaan grup pengguna—yaitu, pengguna dihapus dari grup IAM pengguna mana pun yang menjadi anggotanya
- Semua kata sandi yang terkait dengan pengguna
- Semua kunci akses milik pengguna
- Semua kebijakan inline yang disertakan di pengguna (kebijakan yang diterapkan ke pengguna melalui izin grup pengguna tidak terpengaruh)

> **Note**
>
> IAMmenghapus kebijakan terkelola apa pun yang dilampirkan pada pengguna saat Anda menghapus pengguna, tetapi tidak menghapus kebijakan terkelola.

- MFAPerangkat terkait apa pun

Untuk menghapus IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Pengguna, lalu pilih kotak centang di samping nama pengguna yang ingin Anda hapus.
3. Pilih Hapus di bagian atas halaman.
4. Dalam kotak dialog konfirmasi, masukkan nama pengguna dalam bidang input teks untuk mengonfirmasi penghapusan pengguna. Pilih Hapus.

Menghapus IAM user ()AWS CLI

Berbeda dengan AWS Management Console, ketika Anda menghapus pengguna dengan AWS CLI, Anda harus menghapus item yang dilampirkan ke pengguna secara manual. Prosedur ini menggambarkan proses.

Untuk menghapus pengguna dari akun Anda (AWS CLI)

1. Hapus kata sandi pengguna, jika pengguna memilikinya.

   `aws iam delete-login-profile`
2. Hapus kunci akses pengguna, jika pengguna memilikinya.

   `aws iam list-access-keys` (untuk mencantumkan kunci akses pengguna) dan `aws iam delete-access-key`
3. Hapus sertifikat masuk pengguna. Perhatikan bahwa ketika Anda menghapus kredensial keamanan, akan hilang selamanya dan tidak dapat dipulihkan.

   `aws iam list-signing-certificates` (untuk mencantumkan sertifikat masuk pengguna) dan `aws iam delete-signing-certificate`
4. Hapus kunci SSH publik pengguna, jika pengguna memilikinya.

   `aws iam list-ssh-public-keys`(untuk mencantumkan kunci SSH publik pengguna) dan `aws iam delete-ssh-public-key`
5. Hapus kredensial Git pengguna.

   `aws iam list-service-specific-credentials` (untuk mencantumkan kredensial git pengguna) dan `aws iam delete-service-specific-credential`
6. Nonaktifkan perangkat otentikasi multi-faktor (MFA) pengguna, jika pengguna memilikinya.

`aws iam list-mfa-devices`(untuk mencantumkan MFA perangkat pengguna), `aws iam deactivate-mfa-device` (untuk menonaktifkan perangkat), dan `aws iam delete-virtual-mfa-device` (untuk menghapus MFA perangkat virtual secara permanen)

7. Hapus kebijakan inline pengguna.

   `aws iam list-user-policies` (untuk mencantumkan kebijakan inline bagi pengguna) dan `aws iam delete-user-policy` (untuk menghapus kebijakan)

8. Lepaskan kebijakan terkelola yang melekat pada pengguna.

   `aws iam list-attached-user-policies` (untuk mencantumkan kebijakan terkelola yang melekat pada pengguna) dan `aws iam detach-user-policy` (untuk melepaskan kebijakan)

9. Hapus pengguna dari IAM grup mana pun.

   `aws iam list-groups-for-user`(untuk membuat daftar IAM grup tempat pengguna berada) dan `aws iam remove-user-from-group`

10. Hapus pengguna.

    `aws iam delete-user`

Menonaktifkan pengguna IAM

Anda mungkin perlu menonaktifkan IAM pengguna saat mereka sementara jauh dari perusahaan Anda. Anda dapat meninggalkan kredensi IAM pengguna mereka di tempat dan masih memblokir akses mereka AWS .

Untuk menonaktifkan pengguna, buat dan lampirkan kebijakan untuk menolak akses pengguna. AWS Anda dapat memulihkan akses pengguna nanti.

Berikut adalah dua contoh kebijakan penolakan yang dapat Anda lampirkan ke pengguna untuk menolak akses mereka.

Kebijakan berikut tidak termasuk batas waktu. Anda harus menghapus kebijakan untuk memulihkan akses pengguna.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
```

```
        "Action": "*",
        "Resource": "*"
      }
   ]
}
```

Kebijakan berikut mencakup kondisi yang memulai kebijakan pada 24 Desember 2024 pukul 23.59 (UTC) dan berakhir pada 28 Februari 2025 pukul 23.59 (). UTC

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
}
```

Kontrol akses IAM pengguna ke AWS Management Console

IAMpengguna dengan izin yang masuk ke Anda Akun AWS melalui AWS Management Console dapat mengakses AWS sumber daya Anda. Daftar berikut menunjukkan cara-cara yang dapat Anda berikan kepada IAM pengguna akses ke Akun AWS sumber daya Anda melalui AWS Management Console. Ini juga menunjukkan bagaimana IAM pengguna dapat mengakses fitur AWS akun lain melalui situs AWS web.

> **Note**
>
> Tidak ada biaya untuk digunakanIAM.

The AWS Management Console

Anda membuat kata sandi untuk setiap IAM pengguna yang membutuhkan akses ke file AWS Management Console. Pengguna mengakses konsol melalui halaman Akun AWS login IAM yang

diaktifkan. Untuk informasi tentang mengakses halaman login, lihat Cara masuk di AWS Sign-In Panduan Pengguna. AWS Untuk informasi tentang cara membuat kata sandi, lihat Kelola kata sandi pengguna di AWS.

Anda dapat mencegah IAM pengguna mengakses AWS Management Console dengan menghapus kata sandi mereka. Hal ini mencegah mereka masuk ke dalam AWS Management Console menggunakan kredensi masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows PowerShell AWS API, atau AWS Console Mobile Application.

AWS Sumber daya Anda, seperti EC2 instans Amazon, bucket Amazon S3, dan sebagainya

Bahkan jika IAM pengguna Anda memiliki kata sandi, mereka masih memerlukan izin untuk mengakses AWS sumber daya Anda. Saat Anda membuat IAM pengguna, pengguna tersebut tidak memiliki izin secara default. Untuk memberi IAM pengguna izin yang mereka butuhkan, Anda melampirkan kebijakan kepada mereka. Jika Anda memiliki banyak IAM pengguna yang melakukan tugas yang sama dengan sumber daya yang sama, Anda dapat menetapkan IAM pengguna tersebut ke grup. Kemudian tetapkan izin untuk grup tersebut. Untuk informasi tentang membuat IAM pengguna dan grup, lihatIAMIdentitas . Untuk informasi tentang menggunakan kebijakan untuk mengatur izin, lihat Manajemen akses untuk AWS sumber daya.

AWS Forum Diskusi

Siapa pun dapat membaca tulisan di Forum Diskusi AWS. Pengguna yang ingin memposting pertanyaan atau komentar ke Forum AWS Diskusi dapat melakukannya menggunakan nama pengguna mereka. Saat pertama kali pengguna memposting ke Forum AWS Diskusi, pengguna diminta untuk memasukkan nama panggilan dan alamat email. Hanya pengguna yang dapat menggunakan nama panggilan itu di Forum AWS Diskusi.

Informasi Akun AWS penagihan dan penggunaan Anda

Anda dapat memberi pengguna akses informasi Akun AWS penagihan dan penggunaan Anda. Untuk informasi selengkapnya, lihat Mengontrol Akses ke Informasi Penagihan Anda di Panduan AWS Billing Pengguna.

Informasi Akun AWS profil Anda

Pengguna tidak dapat mengakses informasi Akun AWS profil Anda.

Kredensi Akun AWS keamanan Anda

Pengguna tidak dapat mengakses Akun AWS kredensil keamanan Anda.

> **Note**
>
> IAMkebijakan mengontrol akses terlepas dari antarmuka. Misalnya, Anda dapat memberikan pengguna dengan kata sandi untuk mengakses AWS Management Console. Kebijakan untuk pengguna tersebut (atau grup apa pun yang dimiliki pengguna) akan mengontrol apa yang dapat dilakukan pengguna di AWS Management Console. Atau, Anda bisa memberi pengguna kunci AWS akses untuk melakukan API panggilan ke AWS. Kebijakan ini akan mengontrol tindakan mana yang dapat pengguna hubungi melalui perpustakaan atau klien yang menggunakan access key tersebut untuk autentikasi.

Mengubah izin untuk pengguna IAM

Anda dapat mengubah izin untuk IAM pengguna Akun AWS dengan mengubah keanggotaan grupnya, dengan menyalin izin dari pengguna yang ada, dengan melampirkan kebijakan langsung ke pengguna, atau dengan menetapkan batas izin. Batas izin mengontrol izin maksimum yang dapat dimiliki pengguna. Batas izin adalah AWS fitur lanjutan.

Untuk informasi tentang izin yang Anda butuhkan untuk memodifikasi izin pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Topik

- Lihat akses pengguna
- Menghasilkan kebijakan berdasarkan aktivitas akses pengguna
- Menambahkan izin ke pengguna (konsol)
- Mengubah izin untuk pengguna (konsol)
- Menghapus kebijakan izin dari pengguna (konsol)
- Menghapus batas izin dari pengguna (konsole)
- Menambahkan dan menghapus izin pengguna (AWS CLI atau AWS API)

Lihat akses pengguna

Sebelum mengubah izin pengguna, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya tentang melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghasilkan kebijakan berdasarkan aktivitas akses pengguna

Terkadang Anda dapat memberikan izin kepada IAM entitas (pengguna atau peran) di luar yang mereka butuhkan. Untuk membantu Anda menyaring izin yang Anda berikan, Anda dapat membuat IAM kebijakan yang didasarkan pada aktivitas akses untuk entitas. IAMAccess Analyzer meninjau AWS CloudTrail log Anda dan membuat templat kebijakan yang berisi izin yang telah digunakan oleh entitas dalam rentang tanggal yang ditentukan. Anda dapat menggunakan templat untuk membuat kebijakan terkelola dengan izin berbutir halus, lalu melampirkannya ke entitas. IAM Dengan begitu, Anda hanya memberikan izin yang dibutuhkan pengguna atau peran untuk berinteraksi dengan AWS sumber daya untuk kasus penggunaan spesifik Anda. Untuk mempelajari selengkapnya, lihat Pembuatan kebijakan IAM Access Analyzer.

Menambahkan izin ke pengguna (konsol)

IAMmenawarkan tiga cara untuk menambahkan kebijakan izin ke pengguna:

- Tambahkan pengguna ke grup – Buat pengguna menjadi anggota kelompok. Kebijakan dari grup melekat ke pengguna.
- Salin izin dari pengguna yang sudah ada – Salin semua keanggotaan grup, kebijakan terkelola terlampir, kebijakan inline, dan batasan izin yang ada dari pengguna sumber.
- Lekatkan kebijakan secara langsung ke pengguna – Lekatkan kebijakan terkelola langsung ke pengguna. Untuk pengelolaan izin yang lebih mudah, lampirkan kebijakan Anda ke grup, lalu buat IAM pengguna anggota grup yang sesuai.

> **Important**
>
> Jika pengguna memiliki batas izin, maka Anda tidak dapat menambahkan izin lebih banyak kepada pengguna daripada yang diizinkan oleh batas izin.

Menambahkan izin dengan menambahkan pengguna ke grup

Menambahkan pengguna ke grup akan segera memengaruhi pengguna.

Untuk menambahkan izin ke pengguna dengan menambahkan pengguna ke grup

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Pengguna.
3. Tinjau keanggotaan grup saat ini untuk IAM pengguna di kolom Grup konsol. Jika perlu, tambahkan kolom ke tabel IAM pengguna dengan menyelesaikan langkah-langkah berikut:

   1. Di atas tabel di ujung kanan, pilih simbol pengaturan
      ().
   2. Di kotak dialog Kelola Kolom, pilih kolom Grup. Secara opsional, Anda juga dapat menghapus kotak centang untuk judul kolom apa pun yang tidak ingin Anda tampilkan di tabel IAM pengguna.
   3. Pilih Tutup untuk kembali ke daftar pengguna.

   Kolom Grup akan memberi tahu Anda grup mana yang dimiliki pengguna. Kolom tersebut mencakup nama grup untuk maksimal dua grup. Jika pengguna adalah anggota dari tiga grup atau lebih, dua grup pertama ditampilkan (diurutkan secara alfabetik), dan jumlah keanggotaan grup tambahan disertakan. Misalnya, jika pengguna termasuk dalam Grup A, Grup B, Grup C, dan Grup D, bidang tersebut berisi nilai Grup A, Grup B + 2 lagi. Untuk melihat jumlah total grup yang menjadi milik pengguna, Anda dapat menambahkan kolom Jumlah grup ke tabel IAM pengguna.
4. Pilih nama pengguna yang izinnya ingin Anda modifikasi.
5. Pilih tab Izin, lalu pilih Tambahkan izin. Pilih Tambahkan pengguna ke grup.
6. Pilih kotak centang untuk setiap grup yang ingin Anda gabungkan pengguna. Daftar ini menunjukkan nama masing-masing grup dan kebijakan yang diterima pengguna jika telah menjadi anggota grup tersebut.
7. (Opsional) Sebagai tambahan untuk memilih dari grup yang ada, Anda dapat memilih Buat grupp untuk menentukan grup baru:

   a. Di tab baru, untuk Nama grup pengguna, ketikkan nama untuk grup baru Anda.

   > **Note**
   >
   > Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota. Nama grup dapat berupa kombinasi hingga 128 huruf, digit, dan karakter ini: plus (+), sama dengan (=), koma (,), titik (.), tanda a keong (@), dan tanda hubung (-). Nama harus unik dalam akun. Grup tidak

dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat dua grup bernama TESTGROUPdan testgroup.

b. Pilih satu atau lebih kotak centang untuk kebijakan terkelola yang ingin Anda lampirkan ke grup. Anda juga dapat membuat kebijakan pengelolaan baru dengan memilih Buat kebijakan. Jika Anda melakukannya, kembali ke tab atau jendela browser ini ketika kebijakan baru selesai; pilih Segarkan; lalu pilih kebijakan baru untuk dilekatkan ke grup Anda. Untuk informasi selengkapnya, lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan.

c. Pilih Buat grup pengguna.

d. Kembali ke tab asal, muat ulang daftar grup Anda. Lalu pilih kotak centang untuk grup baru Anda.

8. Pilih Berikutnya untuk melihat daftar keanggotaan grup yang akan ditambahkan ke pengguna. Lalu, pilih Tambahkan izin.

Menambahkan izin dengan menyalin dari pengguna lain

Menyalin izin akan segera memengaruhi pengguna.

Untuk menambahkan izin ke pengguna dengan menyalin izin dari pengguna lain

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Pengguna di panel navigasi, pilih nama pengguna yang izinnya ingin Anda ubah, lalu pilih tab Izin.
3. Pilih Tambahkan izin, lalu pilih Salin izin dari pengguna yang ada. Daftar ini menampilkan IAM pengguna yang tersedia bersama dengan keanggotaan grup dan kebijakan terlampir mereka. Jika daftar lengkap grup atau kebijakan tidak sesuai pada satu baris, Anda dapat memilih tautan untuk dan ***n*** lainnya. Tindakan tersebut membuka tab browser baru dan melihat daftar kebijakan lengkap (Izin) dan grup (tab Grup).
4. Pilih tombol radio di samping pengguna yang izinnya ingin Anda salin.
5. Pilih Berikutnya untuk melihat daftar perubahan yang akan dilakukan pada pengguna. Lalu, pilih Tambahkan izin.

Menambahkan izin dengan melampirkan kebijakan secara langsung ke pengguna

Melampirkan kebijakan akan segera memengaruhi pengguna.

Untuk menambahkan izin kepada pengguna dengan melampirkan secara langsung kebijakan yang dikelola

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Pengguna di panel navigasi, pilih nama pengguna yang izinnya ingin Anda ubah, lalu pilih tab Izin.
3. Pilih Tambahkan izin, lalu pilih Lampirkan kebijakan secara langsung.
4. Pilih satu kotak centang atau lebih untuk kebijakan terkelola yang ingin Anda lampirkan ke pengguna. Anda juga dapat membuat kebijakan pengelolaan baru dengan memilih Buat kebijakan. Jika Anda melakukannya, kembali ke tab atau jendela browser ini jika kebijakan baru sudah selesai. Pilih Segarkan; dan kemudian pilih kotak centang untuk kebijakan baru untuk dilampirkan ke pengguna Anda. Untuk informasi selengkapnya, lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan.
5. Pilih Berikutnya untuk melihat daftar kebijakan yang akan dilampirkan ke pengguna. Lalu, pilih Tambahkan izin.

Mengatur batas izin untuk pengguna

Pengaturan batas izin akan segera memengaruhi pengguna.

Untuk menetapkan batas izin bagi pengguna

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki batas izin yang ingin Anda ubah.
4. Pilih tab Izin. Jika perlu, buka bagian batas Izin dan kemudian pilih Tetapkan batas izin.
5. Pilih kebijakan yang ingin Anda gunakan untuk batasan izin.
6. Pilih Tetapkan batas.

Mengubah izin untuk pengguna (konsol)

IAMmemungkinkan Anda mengubah izin yang terkait dengan pengguna dengan cara berikut:

- Edit kebijakan izin – Edit kebijakan inline pengguna, kebijakan inline grup pengguna, atau edit kebijakan yang terkelola yang dilampirkan pada pengguna secara langsung atau dari grup. Jika pengguna memiliki batas izin, maka Anda tidak dapat memberikan izin lebih dari yang diizinkan oleh kebijakan yang digunakan sebagai batas izin pengguna.
- Mengubah batas izin – Ubah kebijakan yang digunakan sebagai batas izin bagi pengguna. Ini dapat memperluas atau membatasi izin maksimum yang dapat dimiliki pengguna.

Mengedit kebijakan izin yang terlampir pada pengguna

Mengubah izin akan segera memengaruhi pengguna.

Untuk mengedit kebijakan terkelola yang dilampirkan milik pengguna

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki kebijakan izin yang ingin Anda ubah.
4. Pilih tab Izin. Jika perlu, buka bagian Kebijakan izin.
5. Pilih nama kebijakan yang ingin Anda edit untuk melihat perincian tentang kebijakan tersebut. Pilih tab Penggunaan kebijakan untuk melihat entitas lain yang mungkin terpengaruh jika Anda mengedit kebijakan.
6. Pilih Tab Izin dan meninjau izin yang diberikan oleh kebijakan. Lalu, pilih Edit kebijakan.
7. Edit kebijakan dan selesaikan rekomendasi validasi kebijakan apa pun. Untuk informasi selengkapnya, lihat Edit IAM kebijakan.
8. Pilih Tinjau kebijakan, tinjau ringkasan kebijakan, lalu pilih Simpan perubahan.

Mengubah batas izin untuk pengguna

Mengubah batas izin akan segera memengaruhi pengguna.

Untuk mengubah kebijakan yang digunakan untuk mengatur batas izin bagi pengguna

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki batas izin yang ingin Anda ubah.
4. Pilih tab Izin. Jika perlu, buka bagian Batas izin lalu pilih Ubah batas.
5. Pilih kebijakan yang ingin Anda gunakan untuk batasan izin.
6. Pilih Tetapkan batas.

Menghapus kebijakan izin dari pengguna (konsol)

Menghapus kebijakan akan segera memengaruhi pengguna.

Untuk menghapus izin bagi pengguna IAM

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki batas izin yang ingin Anda hapus.
4. Pilih tab Izin.
5. Jika Anda ingin menghapus izin dengan menghapus kebijakan yang ada, lihat Jenis untuk memahami cara pengguna mendapatkan kebijakan tersebut sebelum memilih Hapus untuk menghapus kebijakan:

   - Jika kebijakan berlaku karena keanggotaan grup, maka memilih Hapus menghapus pengguna dari grup. Ingatlah bahwa Anda mungkin memiliki beberapa kebijakan yang terlampir pada satu grup. Jika Anda menghapus pengguna dari grup, pengguna kehilangan akses ke semua kebijakan yang diterima melalui keanggotaan grup tersebut.
   - Jika kebijakan tersebut merupakan kebijakan terkelola yang dilampirkan langsung ke pengguna, maka memilih Hapus akan melepaskan kebijakan dari pengguna. Hal ini tidak memengaruhi kebijakan itu sendiri atau entitas lainnya yang mungkin menyertai kebijakan tersebut.
   - Jika kebijakan tersebut merupakan kebijakan yang disematkan sebaris, maka memilih X akan menghapus kebijakan tersebutIAM. Kebijakan inline yang dilampirkan langsung ke pengguna hanya ada pada pengguna tersebut.

Menghapus batas izin dari pengguna (konsole)

Menghapus batas izin akan segera memengaruhi pengguna.

Untuk menghapus batas izin dari pengguna

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki batas izin yang ingin Anda hapus.
4. Pilih tab Izin. Jika perlu, buka bagian Batas izin lalu pilih Hapus batas.
5. Pilih Hapus batas untuk mengonfirmasi bahwa Anda ingin menghapus batas izin.

Menambahkan dan menghapus izin pengguna (AWS CLI atau AWS API)

Untuk menambah atau menghapus izin secara terprogram, Anda harus menambahkan atau menghapus keanggotaan grup, melampirkan atau melepaskan kebijakan terkelola, atau menambahkan atau menghapus kebijakan inline. Untuk informasi selengkapnya, lihat topik berikut:

- Mengedit pengguna dalam IAM IAM grup
- Menambahkan dan menghapus izin identitas IAM

Kelola kata sandi pengguna di AWS

Anda dapat mengelola kata sandi untuk IAM pengguna di akun Anda. IAMpengguna membutuhkan kata sandi untuk mengakses file AWS Management Console. Pengguna tidak memerlukan kata sandi untuk mengakses AWS sumber daya secara terprogram dengan menggunakan AWS CLI, Alat untuk Windows PowerShell, atau. AWS SDKs APIs Untuk lingkungan tersebut, Anda memiliki opsi untuk menetapkan kunci akses IAM pengguna. Namun, ada alternatif lain yang lebih aman untuk mengakses kunci yang kami sarankan Anda pertimbangkan terlebih dahulu. Untuk informasi selengkapnya, lihat AWS kredensi keamanan.

> **Note**
>
> Jika salah satu IAM pengguna Anda kehilangan atau lupa kata sandi mereka, Anda tidak dapat mengambilnyaIAM. Tergantung pada pengaturan Anda, baik pengguna atau administrator harus membuat kata sandi baru.

Konten

Menetapkan kebijakan kata sandi akun untuk IAM pengguna

Anda dapat menetapkan kebijakan kata sandi khusus Akun AWS untuk menentukan persyaratan kompleksitas dan periode rotasi wajib untuk kata sandi IAM pengguna Anda. Jika Anda tidak menetapkan kebijakan kata sandi khusus, kata sandi IAM pengguna harus memenuhi kebijakan AWS kata sandi default. Untuk informasi selengkapnya, lihat Opsi kebijakan kata sandi kustom.

Topik

Aturan untuk menetapkan kebijakan kata sandi

Kebijakan IAM kata sandi tidak berlaku untuk Pengguna root akun AWS kata sandi atau kunci akses IAM pengguna. Jika kata sandi kedaluwarsa, IAM pengguna tidak dapat masuk AWS Management Console tetapi dapat terus menggunakan kunci aksesnya.

Saat Anda membuat atau mengubah kebijakan kata sandi, sebagian besar pengaturan kebijakan kata sandi diterapkan saat pengguna Anda mengubah kata sandinya. Namun, beberapa pengaturan diterapkan dengan segera. Sebagai contoh:

- Saat persyaratan panjang dan jenis karakter minimum berubah, pengaturan ini diterapkan di lain waktu saat pengguna Anda mengubah kata sandinya. Pengguna tidak dipaksa untuk mengubah kata sandi yang sudah ada, sekalipun kata sandi yang sudah ada tidak mematuhi kebijakan kata sandi yang diperbarui.
- Saat Anda menetapkan periode kedaluwarsa kata sandi, periode kedaluwarsa tersebut akan segera diberlakukan. Misalnya, anggaplah Anda mengatur masa kedaluwarsa kata sandi selama 90 hari. Dalam hal ini, kata sandi kedaluwarsa untuk semua IAM pengguna yang kata sandi yang ada lebih dari 90 hari. Pengguna tersebut harus mengubah kata sandi saat masuk lagi.

Anda tidak dapat membuat “kebijakan penguncian” untuk mengunci pengguna keluar dari akun setelah sejumlah percobaan masuk gagal yang ditentukan. Untuk keamanan yang lebih baik, kami sarankan Anda menggabungkan kebijakan kata sandi yang kuat dengan otentikasi multi-faktor ()MFA. Untuk informasi lebih lanjut tentangMFA, lihatAWS Otentikasi multi-faktor di IAM.

Izin yang diperlukan untuk menetapkan kebijakan kata sandi

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) melihat atau mengedit kebijakan kata sandi akun mereka. Anda dapat menyertakan tindakan kebijakan kata sandi berikut dalam IAM kebijakan:

- `iam:GetAccountPasswordPolicy` – Memungkinkan entitas untuk melihat kebijakan kata sandi untuk akun mereka
- `iam:DeleteAccountPasswordPolicy` – Memungkinkan entitas untuk menghapus kebijakan kata sandi kustom untuk akun mereka dan kembali ke kebijakan kata sandi default
- `iam:UpdateAccountPasswordPolicy` – Memungkinkan entitas untuk membuat atau mengubah kebijakan kata sandi untuk akun mereka

Kebijakan berikut memungkinkan akses penuh untuk melihat dan mengedit kebijakan kata sandi akun. Untuk mempelajari cara membuat IAM kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk informasi tentang izin yang diperlukan bagi IAM pengguna untuk mengubah kata sandi mereka sendiri, lihatIzinkan IAM pengguna untuk mengubah kata sandi mereka sendiri.

Kebijakan kata sandi default

Jika administrator tidak menetapkan kebijakan kata sandi khusus, kata sandi IAM pengguna harus memenuhi kebijakan AWS kata sandi default.

Kebijakan kata sandi default memberlakukan kondisi berikut:

- Panjang kata sandi minimum adalah 8 karakter dan panjang maksimal 128 karakter
- Minimal tiga dari campuran tipe karakter berikut: huruf besar, huruf kecil, angka, dan karakter non-alfanumerik () `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`
- Tidak identik dengan Akun AWS nama atau alamat email Anda
- Jangan pernah kedaluwarsa kata sandi

Opsi kebijakan kata sandi kustom

Saat Anda mengonfigurasi kebijakan kata sandi khusus untuk akun Anda, Anda dapat menentukan kondisi berikut:

- Panjang minimum kata sandi – Anda dapat menentukan minimal 6 karakter dan maksimal 128 karakter.
- Kekuatan kata sandi - Anda dapat memilih salah satu kotak centang berikut untuk menentukan kekuatan kata sandi IAM pengguna Anda:
  - Memerlukan setidaknya satu huruf besar dari alfabet Latin (A—Z)

  - Memerlukan setidaknya satu huruf kecil dari alfabet Latin (a-z)
  - Diperlukan setidaknya satu angka
  - Diperlukan setidaknya satu karakter nonalfanumerik `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`
- Aktifkan kedaluwarsa kata sandi — Anda dapat memilih dan menentukan minimal 1 dan maksimum 1.095 hari bahwa kata sandi IAM pengguna valid setelah disetel. Misalnya, jika Anda menentukan kedaluwarsa 90 hari, itu langsung berdampak pada semua pengguna Anda. Untuk pengguna dengan kata sandi yang lebih tua dari 90 hari, ketika mereka masuk ke konsol setelah perubahan, mereka harus menetapkan kata sandi baru. Pengguna dengan kata sandi berusia 75-89 hari menerima AWS Management Console peringatan tentang kedaluwarsa kata sandi mereka. IAMpengguna dapat mengubah kata sandi mereka kapan saja jika mereka memiliki izin. Saat mereka mengatur kata sandi baru, periode kedaluwarsa kata sandi tersebut dimulai ulang. IAMPengguna hanya dapat memiliki satu kata sandi yang valid dalam satu waktu.
- Kedaluwarsa kata sandi memerlukan pengaturan ulang administrator - Pilih opsi ini untuk mencegah IAM pengguna menggunakan AWS Management Console untuk memperbarui kata sandi mereka sendiri setelah kata sandi kedaluwarsa. Sebelum Anda memilih opsi ini, konfirmasikan bahwa Anda Akun AWS memiliki lebih dari satu pengguna dengan izin administratif untuk mengatur ulang kata sandi IAM pengguna. Administrator dengan `iam:UpdateLoginProfile` izin dapat mengatur ulang kata sandi IAM pengguna. IAMpengguna dengan `iam:ChangePassword` izin dan kunci akses aktif dapat mengatur ulang kata sandi konsol IAM pengguna mereka sendiri secara terprogram. Jika Anda menghapus kotak centang ini, IAM pengguna dengan kata sandi kedaluwarsa masih harus menetapkan kata sandi baru sebelum mereka dapat mengakses. AWS Management Console
- Izinkan pengguna mengubah kata sandi mereka sendiri — Anda dapat mengizinkan semua IAM pengguna di akun Anda untuk mengubah kata sandi mereka sendiri. Ini memberi pengguna akses ke `iam:ChangePassword` tindakan hanya untuk pengguna mereka dan `iam:GetAccountPasswordPolicy` tindakan. Opsi ini tidak melampirkan kebijakan izin untuk setiap pengguna. Sebaliknya, IAM menerapkan izin di tingkat akun untuk semua pengguna. Atau, Anda hanya dapat mengizinkan beberapa pengguna untuk mengelola kata sandi mereka sendiri. Untuk melakukannya, kosongkan kotak centang ini. Untuk informasi selengkapnya tentang penggunaan kebijakan untuk membatasi siapa yang dapat mengelola kata sandi, lihat Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri.
- Mencegah penggunaan kembali kata sandi — Anda dapat mencegah IAM pengguna menggunakan kembali sejumlah kata sandi sebelumnya yang ditentukan. Anda dapat menentukan jumlah minimum 1 dan jumlah maksimum 24 dari kata sandi sebelumnya yang dapat diulang.

Mengatur kebijakan kata sandi (konsol)

Anda dapat menggunakan kebijakan AWS Management Console untuk membuat, mengubah, atau menghapus kata sandi khusus.

Untuk membuat kebijakan kata sandi khusus (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengaturan akun.
3. Di bagian Kebijakan kata sandi, pilih Edit.
4. Pilih Kustom untuk menggunakan kebijakan kata sandi khusus.
5. Pilih opsi yang ingin Anda terapkan ke kebijakan kata sandi Anda dan pilih Simpan perubahan.
6. Konfirmasikan bahwa Anda ingin menetapkan kebijakan kata sandi khusus dengan memilih Setel kustom.

Untuk mengubah kebijakan kata sandi khusus (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengaturan akun.
3. Di bagian Kebijakan kata sandi, pilih Edit.
4. Pilih opsi yang ingin Anda terapkan ke kebijakan kata sandi Anda dan pilih Simpan perubahan.
5. Konfirmasikan bahwa Anda ingin menetapkan kebijakan kata sandi khusus dengan memilih Setel kustom.

Untuk menghapus kebijakan kata sandi khusus (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengaturan akun.
3. Di bagian Kebijakan kata sandi, pilih Edit.
4. Pilih IAMdefault untuk menghapus kebijakan kata sandi khusus dan pilih Simpan perubahan.
5. Konfirmasikan bahwa Anda ingin mengatur kebijakan kata sandi IAM default dengan memilih Set default.

Mengatur kebijakan kata sandi (AWS CLI)

Anda dapat menggunakan AWS Command Line Interface untuk menetapkan kebijakan kata sandi.

Untuk mengelola kebijakan kata sandi akun kustom dari AWS CLI

Jalankan perintah berikut.

- Untuk membuat atau mengubah kebijakan kata sandi kustom: `aws iam update-account-password-policy`
- Untuk melihat kebijakan kata sandi: `aws iam get-account-password-policy`
- Untuk menghapus kebijakan kata sandi khusus: `aws iam delete-account-password-policy`

Mengatur kebijakan kata sandi (AWS API)

Anda dapat menggunakan AWS API operasi untuk menetapkan kebijakan kata sandi.

Untuk mengelola kebijakan kata sandi akun kustom dari AWS API

Hubungi operasi berikut ini:

- Untuk membuat atau mengubah kebijakan kata sandi kustom: `UpdateAccountPasswordPolicy`
- Untuk melihat kebijakan kata sandi: `GetAccountPasswordPolicy`
- Untuk menghapus kebijakan kata sandi khusus: `DeleteAccountPasswordPolicy`

Kelola kata sandi untuk pengguna IAM

IAMpengguna yang menggunakan AWS Management Console untuk bekerja dengan AWS sumber daya harus memiliki kata sandi untuk masuk. Anda dapat membuat, mengubah, atau menghapus kata sandi untuk IAM pengguna di AWS akun Anda.

Setelah Anda menetapkan kata sandi untuk pengguna, pengguna dapat masuk ke AWS Management Console menggunakan login URL untuk akun Anda, yang terlihat seperti ini:

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

Untuk informasi selengkapnya tentang cara IAM pengguna masuk AWS Management Console, lihat Cara masuk AWS di Panduan AWS Sign-In Pengguna.

Sekalipun pengguna Anda memiliki kata sandi, mereka masih memerlukan izin untuk mengakses sumber daya AWS Anda. Secara default, pengguna tidak memiliki izin. Untuk memberi pengguna Anda izin yang mereka butuhkan, Anda menetapkan kebijakan untuk mereka atau ke grup yang mereka miliki. Untuk informasi tentang membuat pengguna dan grup, lihat IAMIdentitas . Untuk informasi tentang menggunakan kebijakan untuk mengatur izin, lihat Mengubah izin untuk pengguna IAM.

Anda dapat memberikan izin kepada pengguna untuk mengubah kata sandi mereka sendiri. Untuk informasi selengkapnya, lihat Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri. Untuk informasi tentang cara pengguna mengakses halaman login akun Anda, lihat Cara masuk AWS di Panduan AWS Sign-In Pengguna.

Topik

- Membuat, mengubah, atau menghapus kata sandi IAM pengguna (konsol)
- Membuat, mengubah, atau menghapus kata sandi IAM pengguna ()AWS CLI
- Membuat, mengubah, atau menghapus kata sandi IAM pengguna ()AWS API

Membuat, mengubah, atau menghapus kata sandi IAM pengguna (konsol)

Anda dapat menggunakan AWS Management Console untuk mengelola kata sandi untuk IAM pengguna Anda.

Ketika pengguna meninggalkan organisasi Anda atau tidak lagi membutuhkan AWS akses, penting untuk menemukan kredensi yang mereka gunakan dan memastikan bahwa mereka tidak lagi beroperasi. Idealnya, Anda menghapus kredensial jika tidak lagi diperlukan. Anda dapat selalu membuat ulang catatan di kemudian hari jika perlu. Setidaknya, Anda harus mengubah kredensial sehingga pengguna sebelumnya tidak lagi memiliki akses.

Untuk menambahkan kata sandi untuk IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang kata sandinya ingin Anda buat.
4. Pilih tab Security credentials, lalu di bawah Console sign-in, pilih Aktifkan akses konsol.
5. Di Aktifkan akses konsol, untuk kata sandi Konsol, pilih apakah akan IAM membuat kata sandi atau membuat kata sandi khusus:

- Untuk IAM membuat kata sandi, pilih Kata sandi yang dibuat otomatis.
- Untuk membuat kata sandi kustom, pilih Kata sandi kustom, dan ketik kata sandi.

> **Note**
>
> Kata sandi yang Anda buat harus memenuhi kebijakan kata sandi akun.

6. Untuk meminta pengguna membuat kata sandi baru saat masuk, pilih Pengguna harus membuat kata sandi baru saat masuk berikutnya. Kemudian pilih Aktifkan akses konsol.

> **Important**
>
> Jika Anda memilih Pengguna harus membuat kata sandi baru pada opsi masuk berikutnya, pastikan bahwa pengguna memiliki izin untuk mengubah kata sandinya. Untuk informasi selengkapnya, lihat Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri.

7. Untuk melihat kata sandi sehingga Anda dapat membagikannya dengan pengguna, pilih Tampilkan di kotak dialog Kata sandi konsol.

> **Important**
>
> Untuk alasan keamanan, Anda tidak dapat mengakses kata sandi setelah menyelesaikan langkah ini, tetapi Anda dapat membuat kata sandi baru kapan saja.

Untuk mengubah kata sandi untuk IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang kata sandinya ingin Anda ubah.
4. Pilih tab Security credentials, lalu di bawah Console sign-in, pilih Kelola akses konsol.
5. Di Kelola akses konsol, pilih Setel ulang kata sandi jika belum dipilih. Jika akses konsol dinonaktifkan, maka tidak diperlukan kata sandi.

6. Untuk akses Konsol, pilih apakah akan IAM membuat kata sandi atau membuat kata sandi khusus:

   - Untuk IAM membuat kata sandi, pilih Kata sandi yang dibuat otomatis.
   - Untuk membuat kata sandi kustom, pilih Kata sandi kustom, dan ketik kata sandi.

   > **Note**
   >
   > Kata sandi yang Anda buat harus memenuhi kebijakan kata sandi, jika saat ini diatur.

7. Untuk meminta pengguna membuat kata sandi baru saat masuk, pilih Pengguna harus membuat kata sandi baru saat masuk berikutnya.

   > **Important**
   >
   > Jika Anda memilih Pengguna harus membuat kata sandi baru pada opsi masuk berikutnya, pastikan bahwa pengguna memiliki izin untuk mengubah kata sandinya. Untuk informasi selengkapnya, lihat Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri.

8. Untuk mencabut sesi konsol aktif pengguna, pilih Cabut sesi konsol aktif. Lalu, pilih Terapkan.

   Saat Anda mencabut sesi konsol aktif untuk pengguna, IAM lampirkan kebijakan sebaris baru ke pengguna yang menolak semua izin untuk semua tindakan. Ini mencakup kondisi yang menerapkan pembatasan hanya jika sesi dibuat sebelum titik waktu ketika Anda mencabut izin, serta sekitar 30 detik ke masa depan. Jika pengguna membuat sesi baru setelah Anda mencabut izin, maka kebijakan penolakan tidak berlaku untuk pengguna tersebut. Jika pengguna mencabut sesi konsol aktif mereka sendiri menggunakan metode ini, mereka akan segera keluar dari AWS Management Console.

   > **Important**
   >
   > Agar berhasil mencabut sesi konsol aktif untuk pengguna, Anda harus memiliki `PutUserPolicy` izin untuk pengguna. Ini memungkinkan Anda untuk melampirkan kebijakan `AWSRevokeOlderSessions` inline kepada pengguna.

9. Untuk melihat kata sandi sehingga Anda dapat membagikannya dengan pengguna, pilih Tampilkan di kotak dialog Kata sandi konsol.

> **Important**
>
> Untuk alasan keamanan, Anda tidak dapat mengakses kata sandi setelah menyelesaikan langkah ini, tetapi Anda dapat membuat kata sandi baru kapan saja.

Untuk menghapus (menonaktifkan) kata sandi IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang kata sandinya ingin Anda hapus.
4. Pilih tab Security credentials, lalu di bawah Console sign-in, pilih Kelola akses konsol.
5. Di Kelola akses konsol, pilih Nonaktifkan akses konsol jika belum dipilih. Jika akses konsol dinonaktifkan, maka tidak diperlukan kata sandi.
6. Untuk mencabut sesi konsol aktif pengguna, pilih Cabut sesi konsol aktif. Kemudian pilih Nonaktifkan akses.

> **Important**
>
> Agar berhasil mencabut sesi konsol aktif untuk pengguna, Anda harus memiliki `PutUserPolicy` izin untuk pengguna. Ini memungkinkan Anda untuk melampirkan kebijakan `AWSRevokeOlderSessions` inline kepada pengguna.

Saat Anda mencabut sesi konsol aktif untuk pengguna, IAM menyematkan kebijakan sebaris baru di IAM pengguna yang menolak semua izin untuk semua tindakan. Ini mencakup kondisi yang menerapkan pembatasan hanya jika sesi dibuat sebelum titik waktu ketika Anda mencabut izin, serta sekitar 30 detik ke masa depan. Jika pengguna membuat sesi baru setelah Anda mencabut izin, maka kebijakan penolakan tidak berlaku untuk pengguna tersebut. Jika pengguna mencabut sesi konsol aktif mereka sendiri menggunakan metode ini, mereka akan segera keluar dari AWS Management Console.

> **Important**
>
> Anda dapat mencegah IAM pengguna mengakses AWS Management Console dengan menghapus kata sandi mereka. Ini mencegah mereka masuk ke AWS Management Console menggunakan kredensi masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows PowerShell AWS API, atau AWS Console Mobile Application.

Membuat, mengubah, atau menghapus kata sandi IAM pengguna ()AWS CLI

Anda dapat menggunakan AWS CLI API untuk mengelola kata sandi untuk IAM pengguna Anda.

Untuk membuat kata sandi (AWS CLI)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: aws iam get-login-profile
2. Untuk membuat kata sandi, jalankan perintah ini: aws iam create-login-profile

Untuk mengubah kata sandi pengguna (AWS CLI)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: aws iam get-login-profile
2. Untuk mengubah kata sandi, jalankan perintah ini: aws iam update-login-profile

Untuk menghapus (menonaktifkan) kata sandi pengguna (AWS CLI)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: aws iam get-login-profile
2. (Opsional) Untuk menentukan kapan kata sandi terakhir digunakan, jalankan perintah ini: aws iam get-user
3. Untuk menghapus kata sandi, jalankan perintah ini: aws iam delete-login-profile

> **Important**
>
> Ketika Anda menghapus kata sandi pengguna, pengguna tidak bisa lagi masuk ke AWS Management Console. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows PowerShell, atau panggilan AWS API fungsi. Saat Anda menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk menghapus pengguna dari Anda Akun AWS, Anda harus terlebih dahulu menghapus kata sandi menggunakan operasi ini. Untuk informasi selengkapnya, lihat Menghapus IAM user ()AWS CLI.

Untuk mencabut sesi konsol aktif pengguna sebelum waktu yang ditentukan ()AWS CLI

1. Untuk menyematkan kebijakan sebaris yang mencabut sesi konsol aktif IAM pengguna sebelum waktu yang ditentukan, gunakan kebijakan sebaris berikut dan jalankan perintah ini: aws iam put-user-policy

   Kebijakan inline ini menolak semua izin dan menyertakan kunci kondisi. `aws:TokenIssueTime` Ini mencabut sesi konsol aktif pengguna sebelum waktu yang ditentukan dalam `Condition` elemen kebijakan sebaris. Ganti nilai kunci `aws:TokenIssueTime` kondisi dengan nilai Anda sendiri.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "*",
       "Resource": "*",
       "Condition": {
         "DateLessThan": {
           "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
         }
       }
     }
   }
   ```

2. (Opsional) Untuk mencantumkan nama kebijakan sebaris yang disematkan di IAM pengguna, jalankan perintah ini: aws iam list-user-policies
3. (Opsional) Untuk melihat kebijakan inline bernama yang disematkan di IAM pengguna, jalankan perintah ini: aws iam get-user-policy

Membuat, mengubah, atau menghapus kata sandi IAM pengguna ()AWS API

Anda dapat menggunakan AWS API untuk mengelola kata sandi untuk IAM pengguna Anda.

Untuk membuat kata sandi (AWS API)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, hubungi operasi ini: GetLoginProfile
2. Untuk membuat kata sandi, hubungi operasi ini: CreateLoginProfile

Untuk mengubah kata sandi pengguna (AWS API)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, hubungi operasi ini: GetLoginProfile
2. Untuk mengubah kata sandi, hubungi operasi ini: UpdateLoginProfile

Untuk menghapus (menonaktifkan) kata sandi pengguna (AWS API)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: GetLoginProfile
2. (Opsional) Untuk menentukan kapan kata sandi terakhir digunakan, jalankan perintah ini: GetUser
3. Untuk menghapus kata sandi, jalankan perintah ini: DeleteLoginProfile

> **Important**
>
> Ketika Anda menghapus kata sandi pengguna, pengguna tidak bisa lagi masuk ke AWS Management Console. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows PowerShell, atau panggilan AWS API fungsi. Saat Anda menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk menghapus pengguna dari Anda Akun AWS, Anda harus terlebih dahulu menghapus kata sandi menggunakan operasi ini. Untuk informasi selengkapnya, lihat Menghapus IAM user ()AWS CLI.

Untuk mencabut sesi konsol aktif pengguna sebelum waktu yang ditentukan ()AWS API

1. Untuk menyematkan kebijakan sebaris yang mencabut sesi konsol aktif IAM pengguna sebelum waktu yang ditentukan, gunakan kebijakan sebaris berikut dan jalankan perintah ini: PutUserPolicy

   Kebijakan inline ini menolak semua izin dan menyertakan kunci kondisi. `aws:TokenIssueTime` Ini mencabut sesi konsol aktif pengguna sebelum waktu yang ditentukan dalam `Condition` elemen kebijakan sebaris. Ganti nilai kunci `aws:TokenIssueTime` kondisi dengan nilai Anda sendiri.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "*",
       "Resource": "*",
       "Condition": {
         "DateLessThan": {
           "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
         }
       }
     }
   }
   ```

2. (Opsional) Untuk mencantumkan nama kebijakan sebaris yang disematkan di IAM pengguna, jalankan perintah ini: ListUserPolicies
3. (Opsional) Untuk melihat kebijakan inline bernama yang disematkan di IAM pengguna, jalankan perintah ini: GetUserPolicy

Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri

> **Note**
>
> Pengguna dengan identitas federasi akan menggunakan proses yang ditentukan oleh penyedia identitas mereka untuk mengubah kata sandi mereka. Sebagai praktik terbaik, mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara.

Anda dapat memberikan izin kepada IAM pengguna untuk mengubah kata sandi mereka sendiri untuk masuk ke AWS Management Console. Anda dapat melakukannya dengan salah satu dari dua cara berikut:

- Izinkan semua IAM pengguna di akun untuk mengubah kata sandi mereka sendiri.
- Izinkan hanya IAM pengguna yang dipilih untuk mengubah kata sandi mereka sendiri. Dalam skenario ini, Anda menonaktifkan opsi bagi semua pengguna untuk mengubah kata sandi mereka sendiri dan Anda menggunakan IAM kebijakan untuk memberikan izin hanya kepada beberapa pengguna. Pendekatan ini memungkinkan pengguna untuk mengubah kartu sandi mereka sendiri dan secara opsional kredensial lainnya seperti access key mereka sendiri.

> **Important**
>
> Kami menyarankan Anda menetapkan kebijakan kata sandi khusus yang mengharuskan IAM pengguna membuat kata sandi yang kuat.

Untuk memungkinkan semua IAM pengguna mengubah kata sandi mereka sendiri

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, klik Pengaturan akun.
3. Di bagian Kebijakan kata sandi, pilih Edit.
4. Pilih Kustom untuk menggunakan kebijakan kata sandi khusus.
5. Pilih Izinkan pengguna mengubah kata sandi mereka sendiri, lalu pilih Simpan perubahan. Hal ini memungkinkan semua pengguna di akun akses ke `iam:ChangePassword` tindakan hanya untuk pengguna mereka dan untuk `iam:GetAccountPasswordPolicy` tindakan.
6. Berikan petunjuk berikut kepada pengguna untuk mengubah kata sandi mereka: Bagaimana pengguna IAM mengubah kata sandi mereka sendiri.

Untuk informasi tentang AWS CLI, Alat untuk Windows PowerShell, dan API perintah yang dapat Anda gunakan untuk mengubah kebijakan kata sandi akun (yang mencakup membiarkan semua pengguna mengubah kata sandi mereka sendiri), lihatMengatur kebijakan kata sandi (AWS CLI).

Untuk memungkinkan IAM pengguna yang dipilih mengubah kata sandi mereka sendiri

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, klik Pengaturan akun.
3. Di bagian Kebijakan kata sandi, pastikan bahwa Perbolehkan pengguna mengubah kata sandi mereka sendiri tidak dipilih. Jika kotak centang ini dipilih, semua pengguna dapat mengubah kata sandi mereka sendiri. (Lihat prosedur sebelumnya.)
4. Buat pengguna yang seharusnya diperbolehkan untuk mengubah kata sandi mereka sendiri, jika mereka belum ada. Untuk detailnya, lihat Buat IAM pengguna di Akun AWS.
5. (Opsional) Buat IAM grup untuk pengguna yang diizinkan untuk mengubah kata sandi mereka, lalu tambahkan pengguna dari langkah sebelumnya ke grup. Untuk detailnya, lihat Grup pengguna IAM.
6. Tetapkan kebijakan berikut kepada grup. Untuk informasi selengkapnya, lihat Kelola IAM kebijakan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

Kebijakan ini memberikan akses ke ChangePasswordtindakan, yang memungkinkan pengguna hanya mengubah kata sandi mereka sendiri dari konsol AWS CLI, Alat untuk Windows PowerShell, atau. API Ini juga memberikan akses ke GetAccountPasswordPolicytindakan, yang memungkinkan pengguna melihat kebijakan kata sandi saat ini; izin ini diperlukan agar pengguna dapat melihat kebijakan kata sandi akun di halaman Ubah kata sandi. Pengguna

harus diizinkan untuk membaca kebijakan kata sandi saat ini untuk memastikan bahwa kata sandi yang diubah memenuhi persyaratan kebijakan.

7. Berikan petunjuk berikut kepada pengguna untuk mengubah kata sandi mereka: Bagaimana pengguna IAM mengubah kata sandi mereka sendiri.

Untuk informasi lebih lanjut

Untuk informasi selengkapnya tentang mengelola kredensial, lihat topik berikut:

- Izinkan IAM pengguna untuk mengubah kata sandi mereka sendiri
- Kelola kata sandi pengguna di AWS
- Menetapkan kebijakan kata sandi akun untuk IAM pengguna
- Kelola IAM kebijakan
- Bagaimana pengguna IAM mengubah kata sandi mereka sendiri

Bagaimana pengguna IAM mengubah kata sandi mereka sendiri

Jika Anda telah diberikan izin untuk mengubah kata sandi pengguna IAM Anda sendiri, Anda dapat menggunakan halaman khusus AWS Management Console untuk melakukan ini. Anda juga dapat menggunakan AWS CLI atau AWS API.

Topik

- Izin diperlukan
- Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)
- Bagaimana pengguna IAM mengubah kata sandi (AWS CLI atau AWS API) mereka sendiri

Izin diperlukan

Untuk mengubah kata sandi untuk pengguna IAM Anda sendiri, Anda harus memiliki izin dari kebijakan berikut: AWS: Memungkinkan pengguna IAM untuk mengubah kata sandi konsol mereka sendiri di halaman kredensi Keamanan.

Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Prosedur berikut menjelaskan bagaimana pengguna IAM dapat menggunakan AWS Management Console untuk mengubah kata sandi mereka sendiri.

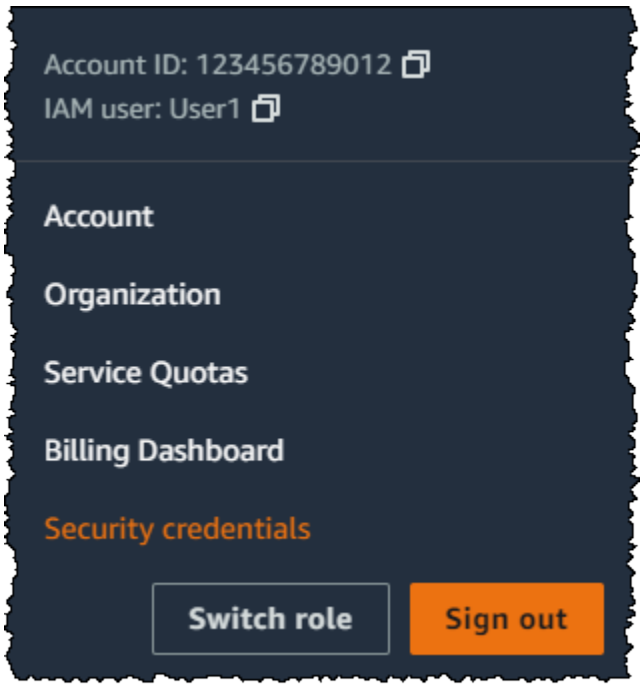
Untuk mengubah kata sandi pengguna IAM Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna IAM dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Pada tab Kredensial AWS IAM, pilih Perbarui kata sandi.
4. Untuk Kata sandi saat ini, masukkan kata sandi Anda saat ini. Masukkan kata sandi baru untuk Kata sandi baru dan Konfirmasi kata sandi baru. Kemudian pilih Perbarui kata sandi.

> **Note**
>
> Kata sandi baru harus memenuhi persyaratan kebijakan kata sandi akun. Untuk informasi selengkapnya, lihat Menetapkan kebijakan kata sandi akun untuk IAM pengguna.

Bagaimana pengguna IAM mengubah kata sandi (AWS CLI atau AWS API) mereka sendiri

Prosedur berikut menjelaskan bagaimana pengguna IAM dapat menggunakan AWS CLI atau AWS API untuk mengubah kata sandi mereka sendiri.

Untuk mengubah kata sandi IAM Anda sendiri, gunakan cara berikut:

- AWS CLI: `aws iam change-password`
- AWS API: `ChangePassword`

Mengelola kunci akses untuk IAM pengguna

Follow us on Twitter

> **Important**
>
> Sebagai praktik terbaik, gunakan kredensi keamanan sementara (seperti IAM peran) alih-alih membuat kredensi jangka panjang seperti kunci akses. Sebelum membuat kunci akses, tinjau alternatif untuk kunci akses jangka panjang.

Kunci akses adalah kredensi jangka panjang untuk IAM pengguna atau. Pengguna root akun AWS Anda dapat menggunakan tombol akses untuk menandatangani permintaan terprogram ke AWS CLI atau AWS API (secara langsung atau menggunakan AWS SDK). Untuk informasi selengkapnya, lihat AWS Signature Version 4 untuk API permintaan.

Access key terdiri dari dua bagian: access key ID (misalnya, `AKIAIOSFODNN7EXAMPLE`) dan secret access key (misalnya, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Anda harus menggunakan ID kunci akses dan kunci akses rahasia bersama-sama untuk mengautentikasi permintaan Anda.

Saat Anda membuat pasangan access key, simpan access key ID dan secret access key di lokasi yang aman. Secret access key hanya tersedia saat Anda membuatnya. Jika Anda kehilangan secret access key, Anda harus menghapus access key dan membuat access key baru. Untuk instruksi lebih lanjut, lihatPerbarui kunci akses.

Anda dapat memiliki maksimal dua kunci akses per pengguna.

> **Important**
>
> Kelola kunci akses Anda dengan aman. Jangan berikan kunci akses Anda kepada pihak yang tidak berwenang, bahkan untuk membantu menemukan pengenal akun Anda. Dengan melakukan tindakan ini, Anda mungkin memberi seseorang akses permanen ke akun Anda.

Topik berikut merinci tugas manajemen yang terkait dengan kunci akses.

Topik

- Izin diperlukan untuk mengelola kunci akses
- Mengelola access key (konsol)
- Mengelola access key (AWS CLI)
- Mengelola access key (AWS API)
- Perbarui kunci akses
- Kunci akses aman
- Gunakan IAM dengan Amazon Keyspaces (untuk Apache Cassandra)

Izin diperlukan untuk mengelola kunci akses

> **Note**
>
> `iam:TagUser`adalah izin opsional untuk menambahkan dan mengedit deskripsi untuk kunci akses. Untuk informasi selengkapnya, silakan lihat Tag IAM pengguna

Untuk membuat kunci akses bagi IAM pengguna Anda sendiri, Anda harus memiliki izin dari kebijakan berikut:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Untuk memperbarui kunci akses untuk IAM pengguna Anda sendiri, Anda harus memiliki izin dari kebijakan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Mengelola access key (konsol)

Anda dapat menggunakan AWS Management Console untuk mengelola kunci akses IAM pengguna.

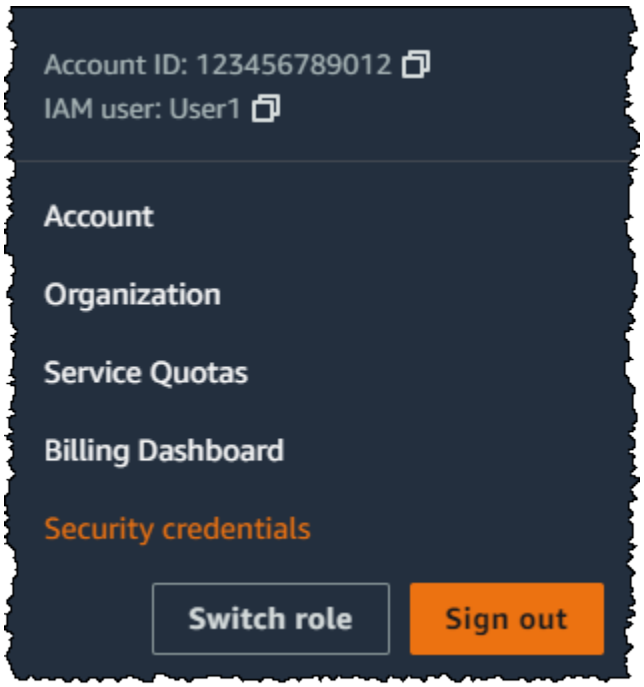
Untuk membuat, memodifikasi, atau menghapus kunci akses Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama IAM pengguna, dan kata sandi Anda untuk masuk ke IAMkonsol.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama IAM pengguna dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login IAM pengguna untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



Lakukan salah satu hal berikut ini:

Untuk membuat kunci akses

1. Pada bagian Access key, pilih Buat access key. Jika Anda sudah memiliki dua kunci akses, tombol ini dinonaktifkan dan Anda harus menghapus kunci akses sebelum Anda dapat membuat yang baru.
2. Pada halaman praktik terbaik & alternatif kunci Access, pilih kasus penggunaan Anda untuk mempelajari opsi tambahan yang dapat membantu Anda menghindari pembuatan kunci akses jangka panjang. Jika Anda menentukan bahwa kasus penggunaan Anda masih memerlukan kunci akses, pilih Lainnya dan kemudian pilih Berikutnya.
3. (Opsional) Tetapkan nilai tag deskripsi untuk kunci akses. Ini menambahkan pasangan nilai kunci tag ke pengguna AndaIAM. Ini dapat membantu Anda mengidentifikasi dan memperbarui kunci akses nanti. Kunci tag diatur ke id kunci akses. Nilai tag diatur ke deskripsi kunci akses yang Anda tentukan. Setelah selesai, pilih Buat kunci akses.
4. Pada halaman Ambil kunci akses, pilih Tampilkan untuk mengungkapkan nilai kunci akses rahasia pengguna Anda, atau Unduh file.csv. Ini adalah satu-satunya kesempatan untuk menyimpan secret access key Anda. Setelah menyimpan kunci akses rahasia di lokasi yang aman, pilih Selesai.

Untuk menonaktifkan kunci akses

- Di bagian Kunci akses temukan kunci yang ingin Anda nonaktifkan, lalu pilih Tindakan, lalu pilih Nonaktifkan. Saat diminta konfirmasi, pilih Deactivate (Nonaktifkan). Access key yang dinonaktifkan masih diperhitungkan dalam batas dua access key Anda.

Untuk mengaktifkan kunci akses

- Di bagian Kunci akses, temukan kunci untuk mengaktifkan, lalu pilih Tindakan, lalu pilih Aktifkan.

Untuk menghapus kunci akses saat Anda tidak lagi membutuhkannya

- Di bagian Kunci akses, cari kunci yang ingin Anda hapus, lalu pilih Tindakan, lalu pilih Hapus. Ikuti instruksi dalam dialog untuk pertama Nonaktifkan dan kemudian konfirmasikan penghapusan. Sebaiknya verifikasi bahwa access key tidak lagi digunakan sebelum Anda menghapusnya secara permanen.

Untuk membuat, memodifikasi, atau menghapus kunci akses IAM pengguna lain (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang access key-nya ingin Anda kelola, lalu pilih tab Kredensial keamanan.
4. Di bagian Tombol akses, lakukan salah satu hal berikut:

   - Untuk membuat access key, pilih Buat access key. Jika tombol dinonaktifkan, maka Anda harus menghapus salah satu kunci yang ada sebelum Anda dapat membuat yang baru. Pada halaman Praktik & Alternatif Terbaik Kunci Akses, tinjau praktik dan alternatif terbaik. Pilih kasus penggunaan Anda untuk mempelajari tentang opsi tambahan yang dapat membantu Anda menghindari membuat kunci akses jangka panjang. Jika Anda menentukan bahwa kasus penggunaan Anda masih memerlukan kunci akses, pilih Lainnya dan kemudian pilih Berikutnya. Pada halaman Retrieve access key, pilih Tampilkan untuk mengungkapkan nilai kunci akses rahasia pengguna Anda. Untuk menyimpan ID kunci akses dan kunci akses rahasia ke `.csv` file ke lokasi aman di komputer Anda, pilih tombol Unduh file.csv. Saat Anda membuat kunci akses untuk pengguna Anda, key pair tersebut aktif secara default, dan pengguna Anda dapat langsung menggunakan pasangan tersebut.
   - Untuk menonaktifkan kunci akses aktif, pilih Tindakan, lalu pilih Nonaktifkan.
   - Untuk mengaktifkan kunci akses tidak aktif, pilih Tindakan, lalu pilih Aktifkan.
   - Untuk menghapus kunci akses Anda, pilih Tindakan, lalu pilih Hapus. Ikuti instruksi dalam dialog untuk pertama Nonaktifkan dan kemudian konfirmasikan penghapusan. AWS merekomendasikan bahwa sebelum Anda melakukan ini, Anda terlebih dahulu menonaktifkan kunci dan menguji bahwa itu tidak lagi digunakan. Saat Anda menggunakan AWS Management Console, Anda harus menonaktifkan kunci Anda sebelum menghapusnya.

Untuk membuat daftar kunci akses untuk IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang dimaksud, lalu pilih tab Kredensial keamanan. Di bagian tombol Akses, Anda akan melihat tombol akses pengguna dan status setiap tombol ditampilkan.

> **Note**
>
> Hanya access key ID pengguna yang terlihat. Secret access key hanya dapat diambil kembali saat kunci dibuat.

Untuk membuat daftar kunci akses IDs untuk beberapa IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan kolom access key ID pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:
   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ( ).
   b. Di Kelola kolom, pilih access key ID.
   c. Pilih Tutup untuk kembali ke daftar pengguna.
4. Kolom access key ID menampilkan setiap access key ID, diikuti oleh statusnya; misalnya, 234782070278420732307623740 23 (Aktif) atau 220937402396702370248434203 27 (Tidak aktif).

   Anda dapat menggunakan informasi ini untuk melihat dan menyalin access key bagi pengguna dengan satu atau dua access key. Kolom menampilkan Tidak ada untuk pengguna yang tidak memiliki access key.

> **Note**
>
> Hanya access key ID dan status pengguna yang terlihat. Secret access key hanya dapat diambil kembali saat kunci dibuat.

Untuk menemukan IAM pengguna mana yang memiliki kunci akses tertentu (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Pengguna.
3. Di kotak pencarian, ketik atau tempelkan access key ID pengguna yang ingin Anda temukan.
4. Jika perlu, tambahkan kolom access key ID pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:

   a. Di atas tabel di ujung kanan, pilih ikon pengaturan
   ().

   b. Di Kelola kolom, pilih access key ID.

   c. Pilih Tutup untuk kembali ke daftar pengguna dan mengonfirmasi bahwa pengguna yang difilter memiliki access key tertentu.

Mengelola access key (AWS CLI)

Untuk mengelola kunci akses IAM pengguna dari AWS CLI, jalankan perintah berikut.

- Untuk membuat access key: `aws iam create-access-key`
- Untuk menonaktifkan atau mengaktifkan kunci akses: `aws iam update-access-key`
- Untuk mencantumkan access key pengguna: `aws iam list-access-keys`
- Untuk menentukan kapan access key digunakan baru-baru ini: `aws iam get-access-key-last-used`
- Untuk menghapus access key: `aws iam delete-access-key`

Mengelola access key (AWS API)

Untuk mengelola kunci akses IAM pengguna dari AWS API, panggil operasi berikut.

- Untuk membuat access key: `CreateAccessKey`
- Untuk menonaktifkan atau mengaktifkan kunci akses: `UpdateAccessKey`
- Untuk mencantumkan access key pengguna: `ListAccessKeys`
- Untuk menentukan kapan access key digunakan baru-baru ini: `GetAccessKeyLastUsed`
- Untuk menghapus access key: `DeleteAccessKey`

Perbarui kunci akses

Sebagai praktik keamanan terbaik, kami menyarankan Anda memperbarui kunci akses IAM pengguna saat diperlukan, seperti ketika seorang karyawan meninggalkan perusahaan Anda. IAMpengguna dapat memperbarui kunci akses mereka sendiri jika mereka telah diberikan izin yang diperlukan.

Untuk detail tentang pemberian izin kepada IAM pengguna untuk memperbarui kunci akses mereka sendiri, lihat. AWS: Memungkinkan pengguna IAM untuk mengelola kata sandi, kunci akses, dan kunci publik SSH mereka sendiri di halaman kredensi Keamanan Anda juga dapat menerapkan kebijakan kata sandi ke akun Anda untuk mengharuskan semua IAM pengguna Anda memperbarui kata sandi mereka secara berkala dan seberapa sering mereka harus melakukannya. Untuk informasi selengkapnya, lihat Menetapkan kebijakan kata sandi akun untuk IAM pengguna.

> **Note**
>
> Gunakan prosedur ini untuk menonaktifkan dan kemudian mengganti kunci akses yang hilang dengan kredensi baru.

Topik

- Memperbarui kunci akses IAM pengguna (konsol)
- Memperbarui kunci akses (AWS CLI)
- Memperbarui kunci akses (AWS API)

Memperbarui kunci akses IAM pengguna (konsol)

Anda dapat memperbarui kunci akses dari file AWS Management Console.

Untuk memperbarui kunci akses bagi IAM pengguna tanpa mengganggu aplikasi Anda (konsol)

1. Meskipun access key pertama masih aktif, buat access key kedua.
    a. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
    b. Di panel navigasi, pilih Pengguna.
    c. Pilih nama pengguna yang dimaksud, lalu pilih tab Kredensial keamanan.

d. Pada bagian Access key, pilih Buat access key. Pada halaman Praktik & alternatif terbaik kunci Akses, pilih Lainnya, lalu pilih Berikutnya.

e. (Opsional) Tetapkan nilai tag deskripsi untuk kunci akses untuk menambahkan pasangan nilai kunci tag ke pengguna iniIAM. Ini dapat membantu Anda mengidentifikasi dan memperbarui kunci akses nanti. Kunci tag diatur ke id kunci akses. Nilai tag diatur ke deskripsi kunci akses yang Anda tentukan. Setelah selesai, pilih Buat kunci akses.

f. Pada halaman Ambil kunci akses, pilih Tampilkan untuk mengungkapkan nilai kunci akses rahasia pengguna Anda, atau Unduh file.csv. Ini adalah satu-satunya kesempatan untuk menyimpan secret access key Anda. Setelah menyimpan kunci akses rahasia di lokasi yang aman, pilih Selesai.

   Saat Anda membuat kunci akses untuk pengguna Anda, key pair tersebut aktif secara default, dan pengguna Anda dapat langsung menggunakan pasangan tersebut. Pada titik ini, pengguna memiliki dua access key aktif.

2. Perbarui semua aplikasi dan alat untuk menggunakan access key baru.
3. Tentukan apakah kunci akses pertama masih digunakan dengan meninjau informasi yang terakhir digunakan untuk kunci akses tertua. Salah satu pendekatannya adalah dengan menunggu beberapa hari kemudian memeriksa access key lama untuk penggunaan apa pun sebelum melanjutkan.
4. Bahkan jika Informasi terakhir yang digunakan menunjukkan bahwa kunci lama tidak pernah digunakan, kami sarankan Anda tidak segera menghapus kunci akses pertama. Sebagai gantinya, pilih Tindakan dan kemudian pilih Nonaktifkan untuk menonaktifkan kunci akses pertama.
5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Aplikasi dan alat apa pun yang masih menggunakan kunci akses asli akan berhenti berfungsi pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengaktifkan kembali kunci akses pertama. Lalu kembali ke Step 3 dan perbarui aplikasi ini untuk menggunakan kunci baru.
6. Setelah Anda menunggu beberapa waktu untuk memastikan bahwa semua aplikasi dan alat telah diperbarui, Anda dapat menghapus access key pertama:

   a. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

   b. Di panel navigasi, pilih Pengguna.

   c. Pilih nama pengguna yang dimaksud, lalu pilih tab Kredensial keamanan.

d. Di bagian Kunci akses untuk kunci akses yang ingin Anda hapus, pilih Tindakan, lalu pilih Hapus. Ikuti instruksi dalam dialog untuk pertama Nonaktifkan dan kemudian konfirmasikan penghapusan.

Untuk menentukan kunci akses mana yang perlu diperbarui atau dihapus (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan kolom Usia access key pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:

   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ().

   b. Di Kelola kolom, pilih Usia access key.

   c. Pilih Tutup untuk kembali ke daftar pengguna.
4. Kolom Usia access key menunjukkan jumlah hari sejak access key aktif tertua dibuat. Anda dapat menggunakan informasi ini untuk menemukan pengguna dengan kunci akses yang mungkin perlu diperbarui atau dihapus. Kolom menampilkan Tidak ada untuk pengguna yang tidak memiliki access key.

Memperbarui kunci akses (AWS CLI)

Anda dapat memperbarui kunci akses dari file AWS Command Line Interface.

Untuk memperbarui kunci akses tanpa mengganggu aplikasi Anda ()AWS CLI

1. Meskipun access key pertama masih aktif, buat access key kedua, yang secara default aktif. Jalankan perintah berikut:

   - `aws iam create-access-key`

     Pada titik ini, pengguna memiliki dua access key aktif.
2. Perbarui semua aplikasi dan alat untuk menggunakan access key baru.
3. Tentukan apakah access key pertama masih digunakan dengan perintah ini:

   - `aws iam get-access-key-last-used`

Salah satu pendekatannya adalah dengan menunggu beberapa hari kemudian memeriksa access key lama untuk penggunaan apa pun sebelum melanjutkan.

4. Bahkan jika langkah Step 3 tidak menunjukkan penggunaan kunci lama, kami sarankan agar Anda tidak segera menghapus access key pertama. Alih-alih, ubah status access key pertama ke `Inactive` menggunakan perintah ini:

   - `aws iam update-access-key`

5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Aplikasi dan alat apa pun yang masih menggunakan kunci akses asli akan berhenti berfungsi pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengubah statusnya kembali `Active` untuk mengaktifkan kembali kunci akses pertama. Lalu kembali ke langkah Step 2 dan perbarui aplikasi ini untuk menggunakan kunci baru.
6. Setelah Anda menunggu beberapa waktu untuk memastikan bahwa semua aplikasi dan alat telah diperbarui, Anda dapat menghapus access key pertama dengan perintah ini:

   - `aws iam delete-access-key`

Memperbarui kunci akses (AWS API)

Anda dapat memperbarui tombol akses menggunakan file AWS API.

Untuk memperbarui kunci akses tanpa mengganggu aplikasi Anda ()AWS API

1. Meskipun access key pertama masih aktif, buat access key kedua, yang secara default aktif. Hubungi operasi berikut ini:

   - `CreateAccessKey`

   Pada titik ini, pengguna memiliki dua access key aktif.
2. Perbarui semua aplikasi dan alat untuk menggunakan access key baru.
3. Tentukan apakah access key pertama masih digunakan dengan menghubungi operasi ini:

   - `GetAccessKeyLastUsed`

Salah satu pendekatannya adalah dengan menunggu beberapa hari kemudian memeriksa access key lama untuk penggunaan apa pun sebelum melanjutkan.

4. Bahkan jika langkah Step 3 tidak menunjukkan penggunaan kunci lama, kami sarankan agar Anda tidak segera menghapus access key pertama. Alih-alih, ubah status access key pertama ke `Inactive` dengan operasi ini:

   - `UpdateAccessKey`

5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Aplikasi dan alat apa pun yang masih menggunakan kunci akses asli akan berhenti berfungsi pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengubah statusnya kembali `Active` untuk mengaktifkan kembali kunci akses pertama. Lalu kembali ke langkah Step 2 dan perbarui aplikasi ini untuk menggunakan kunci baru.
6. Setelah Anda menunggu beberapa waktu untuk memastikan bahwa semua aplikasi dan alat telah diperbarui, Anda dapat menghapus access key pertama dengan memanggil operasi ini:

   - `DeleteAccessKey`

Kunci akses aman

Siapa pun yang memiliki kunci akses Anda memiliki tingkat akses yang sama ke AWS sumber daya Anda seperti yang Anda lakukan. Akibatnya, AWS berupaya keras untuk melindungi kunci akses Anda, dan, sesuai dengan model tanggung jawab bersama kami, Anda juga harus melakukannya.

Perluas bagian berikut untuk panduan untuk membantu Anda melindungi kunci akses Anda.

> **Note**
>
> Organisasi Anda mungkin memiliki persyaratan dan kebijakan keamanan yang berbeda dari yang dijelaskan dalam topik ini. Saran yang diberikan di sini dimaksudkan sebagai pedoman umum.

Hapus (atau jangan buat) kunci Pengguna root akun AWS akses

Salah satu cara terbaik untuk melindungi akun Anda adalah dengan tidak memiliki kunci akses untuk Anda Pengguna root akun AWS. Kecuali Anda harus memiliki kunci akses pengguna root (yang

jarang terjadi), yang terbaik adalah tidak membuatnya. Sebagai gantinya, buat pengguna administratif AWS IAM Identity Center untuk tugas administratif harian.Untuk informasi tentang cara membuat pengguna administratif di Pusat IAM Identitas, lihat Memulai di Panduan Pengguna Pusat IAM Identitas.

Jika Anda sudah memiliki kunci akses pengguna root untuk akun Anda, kami sarankan yang berikut: Temukan tempat di aplikasi Anda di mana Anda saat ini menggunakan kunci akses (jika ada), dan ganti kunci akses pengguna root dengan kunci akses IAM pengguna. Kemudian nonaktifkan dan hapus kunci akses pengguna root. Untuk informasi selengkapnya tentang cara memperbarui kunci akses, lihat Perbarui kunci akses

Gunakan kredensil keamanan sementara (IAMperan) alih-alih kunci akses jangka panjang

Dalam banyak skenario, Anda tidak memerlukan kunci akses jangka panjang yang tidak pernah kedaluwarsa (seperti yang Anda miliki dengan IAM pengguna). Sebagai gantinya, Anda dapat membuat IAM peran dan menghasilkan kredensil keamanan sementara. Kredensial keamanan sementara terdiri dari access key ID dan secret access key, tetapi mereka juga menyertakan token keamanan yang menunjukkan kapan kredensial kedaluwarsa.

Kunci akses jangka panjang, seperti yang terkait dengan IAM pengguna dan pengguna root, tetap valid sampai Anda mencabutnya secara manual. Namun, kredensi keamanan sementara yang diperoleh melalui IAM peran dan fitur lain dari AWS Security Token Service kedaluwarsa setelah periode waktu yang singkat. Gunakan kredensial keamanan sementara untuk membantu mengurangi risiko Anda jika kredensial terekspos secara tidak sengaja.

Gunakan IAM peran dan kredensi keamanan sementara dalam skenario berikut:

- Anda memiliki aplikasi atau AWS CLI skrip yang berjalan di EC2 instans Amazon. Jangan gunakan kunci akses langsung di aplikasi Anda. Jangan memberikan access key ke aplikasi, menyematkannya di aplikasi, atau membiarkan aplikasi membaca access key dari sumber mana pun. Sebagai gantinya, tentukan IAM peran yang memiliki izin yang sesuai untuk aplikasi Anda dan luncurkan instance Amazon Elastic Compute Cloud EC2 (Amazon) dengan peran untuk. EC2 Melakukan hal ini mengaitkan IAM peran dengan EC2 instans Amazon. Praktik ini juga memungkinkan aplikasi untuk mendapatkan kredensi keamanan sementara yang pada gilirannya dapat digunakan untuk melakukan panggilan terprogram. AWS The AWS SDKs and the AWS Command Line Interface (AWS CLI) bisa mendapatkan kredensi sementara dari peran secara otomatis.

- Anda perlu memberikan akses lintas akun. Gunakan IAM peran untuk membangun kepercayaan antar akun, lalu berikan izin terbatas kepada pengguna dalam satu akun untuk mengakses akun tepercaya. Untuk informasi selengkapnya, lihat IAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran.
- Anda memiliki aplikasi seluler. Jangan menyematkan kunci akses dengan aplikasi, bahkan di penyimpanan terenkripsi. Sebagai gantinya, gunakan Amazon Cognito untuk mengelola identitas pengguna di aplikasi Anda. Layanan ini memungkinkan Anda mengautentikasi pengguna menggunakan Login with Amazon, Facebook, Google, atau penyedia identitas yang kompatibel dengan OpenID OIDC Connect (). Anda kemudian dapat menggunakan penyedia kredensial Amazon Cognito untuk mengelola kredensial yang digunakan aplikasi Anda untuk membuat permintaan ke AWS.
- Anda ingin bergabung AWS dan organisasi Anda mendukung SAML 2.0. Jika Anda bekerja untuk organisasi yang memiliki penyedia identitas yang mendukung SAML 2.0, konfigurasikan penyedia untuk digunakanSAML. Anda dapat menggunakan SAML untuk bertukar informasi otentikasi dengan AWS dan mendapatkan kembali satu set kredensi keamanan sementara. Untuk informasi selengkapnya, lihat SAML2.0 federasi.
- Anda ingin bergabung ke dalam AWS dan organisasi Anda memiliki toko identitas lokal. Jika pengguna dapat mengautentikasi di dalam organisasi Anda, Anda dapat menulis aplikasi yang dapat mengeluarkan kredensi keamanan sementara untuk akses ke sumber daya. AWS Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

> **Note**
>
> Apakah Anda menggunakan EC2 instans Amazon dengan aplikasi yang memerlukan akses terprogram ke AWS sumber daya? Jika demikian, gunakan IAMperan untuk EC2.

Kelola kunci akses IAM pengguna dengan benar

Jika Anda harus membuat kunci akses untuk akses terprogram AWS, buat untuk IAM pengguna, berikan pengguna hanya izin yang mereka butuhkan.

Perhatikan tindakan pencegahan ini untuk membantu melindungi kunci akses IAM pengguna:

- Jangan menanamkan kunci akses langsung ke kode. Alat AWS SDKsdan Baris AWS Perintah memungkinkan Anda untuk menempatkan kunci akses di lokasi yang diketahui sehingga Anda tidak perlu menyimpannya dalam kode.

Letakkan access key di salah satu lokasi berikut:

- File AWS kredensialnya. Itu AWS SDKs dan secara AWS CLI otomatis menggunakan kredensil yang Anda simpan di file AWS kredensial.

  Untuk informasi tentang menggunakan file AWS kredensial, lihat dokumentasi untuk file Anda. SDK Contohnya termasuk Set AWS Credentials dan Region dalam Panduan AWS SDK for Java Pengembang dan Konfigurasi dan file kredensi di Panduan Pengguna.AWS Command Line Interface

  Untuk menyimpan kredensil untuk AWS SDK for .NET dan AWS Tools for Windows PowerShell, kami sarankan Anda menggunakan Store. SDK Untuk informasi selengkapnya, lihat Menggunakan SDK Toko di Panduan AWS SDK for .NET Pengembang.

- Variabel lingkungan. Pada sistem multi-tenant, pilih variabel lingkungan pengguna, bukan variabel lingkungan sistem.

  Untuk informasi selengkapnya tentang menggunakan variabel lingkungan untuk menyimpan kredensial, lihat Variabel Lingkungan di Panduan Pengguna AWS Command Line Interface .

- Gunakan tombol akses yang berbeda untuk aplikasi yang berbeda. Lakukan ini sehingga Anda dapat mengisolasi izin dan mencabut kunci akses untuk aplikasi individual jika diekspos. Memiliki kunci akses terpisah untuk aplikasi yang berbeda juga menghasilkan entri yang berbeda dalam file AWS CloudTraillog. Konfigurasi ini memudahkan Anda untuk menentukan aplikasi mana yang melakukan tindakan tertentu.
- Perbarui tombol akses bila diperlukan. Jika ada risiko bahwa kunci akses dapat dikompromikan, perbarui kunci akses dan hapus kunci akses sebelumnya. Untuk detailnya, lihat Perbarui kunci akses
- Hapus kunci akses yang tidak digunakan. Jika pengguna meninggalkan organisasi Anda, hapus IAM pengguna yang sesuai sehingga pengguna tidak dapat lagi mengakses sumber daya Anda. Untuk mengetahui kapan kunci akses terakhir digunakan, gunakan `GetAccessKeyLastUsed`API(AWS CLI perintah: `aws iam get-access-key-last-used`).
- Gunakan kredensi sementara dan konfigurasikan otentikasi multi-faktor untuk operasi Anda yang paling sensitif. API Dengan IAM kebijakan, Anda dapat menentukan API operasi mana yang diizinkan untuk dipanggil oleh pengguna. Dalam beberapa kasus, Anda mungkin menginginkan keamanan tambahan yang mengharuskan pengguna diautentikasi AWS MFA sebelum Anda mengizinkan mereka melakukan tindakan yang sangat sensitif. Misalnya, Anda mungkin memiliki kebijakan yang memungkinkan pengguna menjalankan Amazon EC2 `RunInstances`, `DescribeInstances`, dan `StopInstances` tindakan. Tetapi Anda mungkin

ingin membatasi tindakan destruktif seperti `TerminateInstances` dan memastikan bahwa pengguna dapat melakukan tindakan itu hanya jika mereka mengautentikasi dengan perangkat. AWS MFA Untuk informasi selengkapnya, lihat APIAkses aman dengan MFA.

Akses aplikasi seluler menggunakan tombol AWS akses

Anda dapat mengakses serangkaian AWS layanan dan fitur terbatas menggunakan aplikasi AWS seluler. Aplikasi seluler membantu Anda mendukung respons insiden saat bepergian. Untuk informasi selengkapnya dan untuk mengunduh aplikasi, lihat AWS Console Mobile Application.

Anda dapat masuk ke aplikasi seluler menggunakan kata sandi konsol atau access key Anda. Sebagai praktik terbaik, jangan gunakan access key pengguna root. Sebagai gantinya, kami sangat menyarankan bahwa selain menggunakan kata sandi atau kunci biometrik pada perangkat seluler Anda, Anda membuat IAM pengguna khusus untuk mengelola AWS sumber daya menggunakan aplikasi seluler. Jika Anda kehilangan perangkat seluler, Anda dapat menghapus akses IAM pengguna.

Untuk masuk menggunakan access key (aplikasi seluler)

1. Buka aplikasi di perangkat seluler Anda.
2. Jika ini pertama kalinya Anda menambahkan identitas ke perangkat, pilih Add an identity (Tambahkan identitas), lalu pilih Access key.

   Jika Anda telah masuk menggunakan identitas lain, pilih ikon menu dan pilih Switch identity (Ganti identitas). Kemudian pilih Sign in as a different identity (Masuk sebagai identitas yang berbeda), lalu Access key.
3. Di halaman Access key, masukkan informasi Anda:

   - Access key ID – Masukkan access key ID Anda.
   - Secret access key – Masukkan secret access key Anda.
   - Nama identitas – Masukkan nama identitas yang akan muncul di aplikasi seluler. Ini tidak perlu cocok dengan nama IAM pengguna Anda.
   - Identitas PIN — Buat nomor identifikasi pribadi (PIN) yang akan Anda gunakan untuk login di masa mendatang.

> **Note**
>
> Jika Anda mengaktifkan biometrik untuk aplikasi AWS seluler, Anda akan diminta untuk menggunakan sidik jari atau pengenalan wajah untuk verifikasi, bukan. PIN Jika biometrik gagal, Anda mungkin diminta untuk melakukannya. PIN

4. Pilih Verify and add keys (Verifikasi dan tambahkan kunci).

   Sekarang Anda dapat mengakses set sumber daya tertentu menggunakan aplikasi seluler.

Informasi terkait

Topik berikut memberikan panduan untuk menyiapkan AWS SDKs dan AWS CLI menggunakan kunci akses:

- Menetapkan AWS kredensial dan Wilayah dalam Panduan Pengembang AWS SDK for Java
- Menggunakan SDK Toko di Panduan AWS SDK for .NET Pengembang
- Memberikan Kredensi ke SDK dalam Panduan Pengembang AWS SDK for PHP
- Konfigurasi dalam dokumentasi Boto 3 (AWS SDKuntuk Python)
- Menggunakan AWS Kredensial di Panduan Pengguna AWS Tools for Windows PowerShell
- File konfigurasi dan kredensi di AWS Command Line Interface Panduan Pengguna
- Memberikan akses menggunakan IAM peran dalam Panduan AWS SDK for .NET Pengembang
- Konfigurasikan IAM peran untuk Amazon EC2 di AWS SDK for Java 2.x

Mengaudit access key

Anda dapat meninjau kunci AWS akses dalam kode Anda untuk menentukan apakah kunci tersebut berasal dari akun yang Anda miliki. Anda dapat meneruskan ID kunci akses menggunakan `aws sts get-access-key-info` AWS CLI perintah atau `GetAccessKeyInfo` AWS APIoperasi.

AWS APIOperasi AWS CLI dan mengembalikan ID dari Akun AWS kunci akses milik. Kunci akses yang IDs dimulai dengan `AKIA` adalah kredensil jangka panjang untuk IAM pengguna atau. Pengguna root akun AWS Kunci akses yang IDs dimulai dengan `ASIA` adalah kredensil sementara yang dibuat menggunakan AWS STS operasi. Jika akun dalam tanggapan ini adalah milik Anda, Anda dapat masuk sebagai pengguna utama dan meninjau kunci akses pengguna utama Anda. Kemudian, Anda dapat menarik laporan kredensil untuk mempelajari IAM pengguna mana yang

memiliki kunci. Untuk mengetahui siapa yang meminta kredensi sementara untuk kunci `ASIA` akses, lihat AWS STS peristiwa di log Anda CloudTrail .

Untuk tujuan keamanan, Anda dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan AWS. Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan identias saat mereka mengasumsikan sebuah peran. Misalnya, Anda dapat meminta IAM pengguna menentukan nama pengguna mereka sendiri sebagai identitas sumber mereka. Ini dapat membantu Anda menentukan pengguna mana yang melakukan tindakan tertentu di AWS. Untuk informasi selengkapnya, lihat `sts:SourceIdentity`.

Operasi ini tidak menunjukkan status access key. Kuncinya mungkin aktif, tidak aktif, atau dihapus. Kunci aktif mungkin tidak memiliki izin untuk melakukan operasi. Memberikan access key yang dihapus mungkin akan kembali sebagai kesalahan bahwa kunci tidak ada.

Gunakan IAM dengan Amazon Keyspaces (untuk Apache Cassandra)

Amazon Keyspaces (untuk Apache Cassandra) adalah layanan basis data yang kompatibel dengan Apache Cassandra yang dapat diskalakan, selalu tersedia, dan terkelola. Anda dapat mengakses Amazon Keyspaces melalui AWS Management Console, atau secara terprogram. Untuk mengakses Amazon Keyspaces secara terprogram dengan kredenal khusus layanan, Anda dapat menggunakan atau driver Cassandra sumber terbuka. `cqlsh` Kredensi khusus layanan mencakup nama pengguna dan kata sandi seperti yang digunakan Cassandra untuk otentikasi dan manajemen akses. Anda dapat memiliki maksimal dua set kredensi khusus layanan untuk setiap layanan yang didukung per pengguna.

Untuk mengakses Amazon Keyspaces secara terprogram dengan kunci AWS akses, Anda dapat menggunakan driver Cassandra AWS SDK, AWS Command Line Interface (AWS CLI) atau sumber terbuka dengan plugin SiGv4. Untuk mempelajari selengkapnya, lihat Membuat dan mengonfigurasi AWS kredensional untuk Amazon Keyspaces di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

> **Note**
>
> Jika Anda berencana untuk berinteraksi dengan Amazon Keyspaces hanya melalui konsol, Anda tidak perlu membuat kredensial khusus layanan. Untuk informasi selengkapnya, lihat Mengakses Amazon Keyspaces menggunakan konsol di Amazon Keyspaces (untuk Apache Cassandra) Panduan Pengembang.

Untuk informasi lebih lanjut tentang izin yang diperlukan untuk mengakses Amazon Keyspaces, lihat Contoh Kebijakan Berbasis Identitas Amazon Keyspaces (untuk Apache Cassandra) dalam Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

Membuat kredensial Amazon Keyspaces (konsol)

Anda dapat menggunakan AWS Management Console untuk menghasilkan kredensi Amazon Keyspaces (untuk Apache Cassandra) untuk pengguna Anda. IAM

Untuk menghasilkan kredensial khusus layanan Amazon Keyspaces (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna lalu pilih nama pengguna yang memerlukan kredensial.
3. Di tab Kredensial Keamanan di bawah Kredensial untuk Amazon Keyspaces (untuk Apache Cassandra), pilih Buat kredensial.
4. Kredensial layanan khusus Anda sekarang tersedia. Ini adalah satu-satunya waktu untuk melihat atau mengunduh kata sandi. Anda tidak dapat memulihkannya nanti. Namun, Anda dapat mengatur ulang kata sandi Anda kapan saja. Simpan pengguna dan kata sandi di lokasi aman, karena Anda akan membutuhkannya nanti.

Menghasilkan kredensi Amazon Keyspaces ()AWS CLI

Anda dapat menggunakan AWS CLI untuk menghasilkan kredensi Amazon Keyspaces (untuk Apache Cassandra) untuk pengguna Anda. IAM

Untuk menghasilkan kredensi khusus layanan Amazon Keyspaces ()AWS CLI

- Gunakan perintah berikut ini.
  - aws iam create-service-specific-credential

Menghasilkan kredensi Amazon Keyspaces ()AWS API

Anda dapat menggunakan AWS API untuk menghasilkan kredensi Amazon Keyspaces (untuk Apache Cassandra) untuk pengguna Anda. IAM

Untuk menghasilkan kredensi khusus layanan Amazon Keyspaces ()AWS API

- Selesaikan operasi berikut:
  - CreateServiceSpecificCredential

AWS Otentikasi multi-faktor di IAM

Untuk meningkatkan keamanan, sebaiknya Anda mengonfigurasi otentikasi multi-faktor (MFA) untuk membantu melindungi sumber daya Anda AWS . Anda dapat mengaktifkan MFA untuk Pengguna root akun AWS dan IAM pengguna. Ketika Anda mengaktifkan MFA untuk pengguna root, itu hanya mempengaruhi kredenal pengguna root. IAMpengguna di akun adalah identitas yang berbeda dengan kredensialnya sendiri, dan setiap identitas memiliki konfigurasi sendiri. MFA

Anda Pengguna root akun AWS dan IAM pengguna dapat mendaftarkan hingga delapan MFA perangkat jenis apa pun. Mendaftarkan beberapa MFA perangkat dapat memberikan fleksibilitas dan membantu Anda mengurangi risiko gangguan akses jika perangkat hilang atau rusak. Anda hanya perlu satu MFA perangkat untuk masuk ke AWS Management Console atau membuat sesi melalui AWS CLI.

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensil sementara saat mengakses. AWS Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat IAM Identitas untuk mengelola akses ke beberapa secara terpusat Akun AWS dan memberi pengguna akses masuk tunggal yang MFA dilindungi ke semua akun yang ditetapkan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di Pusat IAM Identitas atau dengan mudah terhubung ke penyedia identitas kompatibel SAML 2.0 yang ada. Untuk informasi lebih lanjut, lihat Apa itu Pusat IAM Identitas? dalam AWS IAM Identity Center User Guide.

MFAmenambahkan keamanan ekstra yang mengharuskan pengguna untuk memberikan otentikasi unik dari MFA mekanisme yang AWS didukung selain kredensi masuk mereka saat mereka mengakses situs AWS web atau layanan.

MFAjenis

AWS mendukung MFA jenis berikut:

Daftar Isi

Kunci sandi dan kunci keamanan

AWS Identity and Access Management mendukung kunci sandi dan kunci keamanan untukMFA. Berdasarkan FIDO standar, kunci sandi menggunakan kriptografi kunci publik untuk memberikan otentikasi yang kuat dan tahan phishing yang lebih aman daripada kata sandi. AWS mendukung dua jenis kunci sandi: kunci sandi terikat perangkat (kunci keamanan) dan kunci sandi yang disinkronkan.

- Kunci keamanan: Ini adalah perangkat fisik, seperti YubiKey, digunakan sebagai faktor kedua untuk otentikasi. Satu kunci keamanan dapat mendukung beberapa akun pengguna root dan IAM pengguna.
- Kunci sandi yang disinkronkan: Ini menggunakan pengelola kredensi dari penyedia seperti Google, Apple, akun Microsoft, dan layanan pihak ketiga seperti 1Password, Dashlane, dan Bitwarden sebagai faktor kedua.

Anda dapat menggunakan autentikator biometrik bawaan, seperti Touch ID di Apple MacBooks, untuk membuka kunci pengelola kredensi dan masuk. AWS Kunci sandi dibuat dengan penyedia pilihan Anda menggunakan sidik jari, wajah, atau perangkat PIN Anda. Anda dapat menyinkronkan kunci sandi di seluruh perangkat Anda untuk memfasilitasi masuk dengan AWS, meningkatkan kegunaan dan pemulihan.

IAMtidak mendukung pendaftaran passkey lokal untuk Windows Hello. Untuk membuat dan menggunakan kunci sandi, pengguna Windows harus menggunakan otentikasi lintas perangkat (). CDA Anda dapat menggunakan CDA kunci sandi dari satu perangkat, seperti perangkat seluler atau kunci keamanan perangkat keras, untuk masuk di perangkat lain seperti laptop.

FIDOAliansi menyimpan daftar semua produk FIDO Bersertifikat yang kompatibel dengan FIDO spesifikasi.

Untuk informasi selengkapnya tentang mengaktifkan kunci sandi dan kunci keamanan, lihat. Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)

Aplikasi otentikator virtual

Aplikasi otentikator virtual berjalan di telepon atau perangkat lain dan mengemulasi perangkat fisik. Aplikasi otentikator virtual mengimplementasikan algoritma kata sandi satu kali (TOTP) berbasis waktu dan mendukung beberapa token pada satu perangkat. Pengguna harus mengetikkan kode yang valid dari perangkat saat diminta saat masuk. Setiap token yang ditetapkan untuk pengguna harus unik. Pengguna tidak dapat mengetik kode dari token pengguna lain untuk mengautentikasi.

Kami menyarankan Anda menggunakan MFA perangkat virtual sambil menunggu persetujuan pembelian perangkat keras atau saat Anda menunggu perangkat keras Anda tiba. Untuk daftar beberapa aplikasi yang didukung yang dapat Anda gunakan sebagai MFA perangkat virtual, lihat Autentikasi Multi-Faktor () MFA.

Untuk petunjuk tentang pengaturan MFA perangkat virtual untuk IAM pengguna, lihatTetapkan MFA perangkat virtual di AWS Management Console.

TOTPToken perangkat keras

Perangkat keras menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali () berbasis waktu. TOTP Pengguna harus mengetik kode yang valid dari perangkat di halaman web kedua saat masuk.

Token ini digunakan secara eksklusif dengan Akun AWS. Anda hanya dapat menggunakan token yang memiliki benih token unik mereka dibagikan dengan AWS aman. Benih token adalah kunci rahasia yang dihasilkan pada saat produksi token. Token yang dibeli dari sumber lain tidak akan berfungsiIAM. Untuk memastikan kompatibilitas, Anda harus membeli MFA perangkat keras Anda dari salah satu tautan berikut: OTPtoken atau kartu OTP tampilan.

- Setiap MFA perangkat yang ditetapkan untuk pengguna harus unik. Pengguna tidak dapat mengetik kode dari perangkat pengguna lain untuk diautentikasi. Untuk informasi tentang MFA perangkat keras yang didukung, lihat Otentikasi Multi-Faktor () MFA.
- Jika Anda ingin menggunakan MFA perangkat fisik, kami sarankan Anda menggunakan kunci keamanan sebagai alternatif TOTP perangkat keras. Kunci keamanan tidak memiliki persyaratan baterai, tahan phishing, dan mendukung banyak pengguna pada satu perangkat.

Anda dapat mengaktifkan kunci sandi atau kunci keamanan dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API. Sebelum Anda dapat mengaktifkan kunci keamanan, Anda harus memiliki akses fisik ke perangkat.

Untuk petunjuk tentang menyiapkan TOTP token perangkat keras untuk IAM pengguna, lihatTetapkan TOTP token perangkat keras di AWS Management Console.

> **Note**
>
> SMSberbasis pesan teks MFA — AWS mengakhiri dukungan untuk mengaktifkan otentikasi SMS multi-faktor (). MFA Kami menyarankan agar pelanggan yang memiliki IAM pengguna yang menggunakan pesan SMS teks berbasis MFA beralih ke salah satu metode alternatif berikut: Kunci sandi atau kunci keamanan, perangkat virtual (berbasis perangkat lunak) MFA, atau perangkat keras. MFA Anda dapat mengidentifikasi pengguna di akun Anda dengan SMS MFA perangkat yang ditetapkan. Di IAM konsol, pilih Pengguna dari panel navigasi, dan cari pengguna dengan SMSdi MFAkolom tabel.

MFArekomendasi

Untuk membantu mengamankan AWS identitas Anda, ikuti rekomendasi ini untuk MFA otentikasi.

- Kami menyarankan Anda mengaktifkan beberapa MFA perangkat untuk Pengguna root akun AWS dan IAM pengguna di Anda Akun AWS. Ini memungkinkan Anda untuk meningkatkan bilah keamanan di Anda Akun AWS dan menyederhanakan pengelolaan akses ke pengguna yang sangat istimewa, seperti. Pengguna root akun AWS
- Anda dapat mendaftarkan hingga delapan MFA perangkat dari kombinasi MFAjenis yang saat ini didukung dengan Anda Pengguna root akun AWS dan IAM pengguna. Dengan beberapa MFA perangkat, Anda hanya perlu satu MFA perangkat untuk masuk ke AWS Management Console atau membuat sesi melalui AWS CLI sebagai pengguna tersebut. IAMPengguna harus mengautentikasi dengan MFA perangkat yang ada untuk mengaktifkan atau menonaktifkan MFA perangkat tambahan.
- Jika MFA perangkat hilang, dicuri, atau tidak dapat diakses, Anda dapat menggunakan salah satu MFA perangkat yang tersisa untuk mengakses Akun AWS tanpa melakukan prosedur Akun AWS pemulihan. Jika MFA perangkat hilang atau dicuri, itu harus dipisahkan dari IAM prinsipal yang terkait dengannya.
- Penggunaan beberapa MFAs memungkinkan karyawan Anda di lokasi yang tersebar secara geografis atau bekerja dari jarak jauh untuk menggunakan berbasis perangkat keras MFA untuk

mengakses AWS tanpa harus mengoordinasikan pertukaran fisik perangkat keras tunggal antara karyawan.

- Penggunaan MFA perangkat tambahan untuk IAM prinsipal memungkinkan Anda untuk menggunakan satu atau lebih MFAs untuk penggunaan sehari-hari, sementara juga menjaga MFA perangkat fisik di lokasi fisik yang aman seperti lemari besi atau aman untuk cadangan dan redundansi.

> **Catatan**
>
> - Anda tidak dapat meneruskan MFA informasi untuk kunci FIDO keamanan ke AWS STS API operasi untuk meminta kredensi sementara.
> - Anda tidak dapat menggunakan AWS CLI perintah atau AWS API operasi untuk mengaktifkan kunci FIDO keamanan.
> - Anda tidak dapat menggunakan nama yang sama untuk lebih dari satu root atau IAM MFA perangkat.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut IAMMFA.

- Untuk informasi selengkapnya tentang penggunaan MFA untuk mengakses AWS, lihatMFAmengaktifkan login.
- Anda dapat memanfaatkan Pusat IAM Identitas untuk mengaktifkan MFA akses aman ke portal AWS akses, aplikasi terintegrasi Pusat IAM Identitas, dan aplikasi AWS CLI. Untuk informasi selengkapnya, lihat Mengaktifkan MFA di Pusat IAM Identitas.

Tetapkan kunci sandi atau kunci keamanan di AWS Management Console

Kunci sandi adalah jenis perangkat otentikasi multi-faktor (MFA) yang dapat Anda gunakan untuk melindungi sumber daya Anda. AWS AWS mendukung kunci sandi yang disinkronkan dan kunci sandi terikat perangkat yang juga dikenal sebagai kunci keamanan.

Kunci sandi yang disinkronkan memungkinkan IAM pengguna mengakses kredensi FIDO masuk mereka di banyak perangkat mereka, bahkan yang baru, tanpa harus mendaftarkan ulang setiap perangkat di setiap akun. Kunci sandi yang disinkronkan mencakup manajer kredensi pihak

pertama seperti Google, Apple, dan Microsoft dan manajer kredensi pihak ketiga seperti 1Password, Dashlane, dan Bitwarden sebagai faktor kedua. Anda juga dapat menggunakan biometrik pada perangkat (misalnya, TouchID, FaceID) untuk membuka kunci pengelola kredensi pilihan Anda untuk menggunakan kunci sandi.

Atau, kunci sandi terikat perangkat terikat ke kunci FIDO keamanan yang Anda colokkan ke USB port di komputer Anda dan kemudian ketuk saat diminta untuk menyelesaikan proses masuk dengan aman. Jika Anda sudah menggunakan kunci FIDO keamanan dengan layanan lain, dan memiliki konfigurasi yang AWS didukung (misalnya, Seri YubiKey 5 dari Yubico), Anda juga dapat menggunakannya dengan. AWS Jika tidak, Anda perlu membeli kunci FIDO keamanan jika Anda ingin menggunakannya WebAuthn untuk MFA masuk AWS. Selain itu, kunci FIDO keamanan dapat mendukung beberapa IAM atau pengguna root pada perangkat yang sama, meningkatkan utilitas mereka untuk keamanan akun. Untuk spesifikasi dan informasi pembelian untuk kedua jenis perangkat, lihat Autentikasi Multi-Faktor.

Anda dapat mendaftarkan hingga delapan MFA perangkat dari kombinasi MFAjenis yang saat ini didukung dengan Anda Pengguna root akun AWS dan IAM pengguna. Dengan beberapa MFA perangkat, Anda hanya perlu satu MFA perangkat untuk masuk ke AWS Management Console atau membuat sesi melalui AWS CLI sebagai pengguna tersebut. Kami menyarankan Anda mendaftarkan beberapa MFA perangkat. Misalnya, Anda dapat mendaftarkan autentikator bawaan dan juga mendaftarkan kunci keamanan yang Anda simpan di lokasi yang aman secara fisik. Jika Anda tidak dapat menggunakan autentikator bawaan Anda, maka Anda dapat menggunakan kunci keamanan terdaftar Anda. Untuk aplikasi autentikator, kami juga menyarankan untuk mengaktifkan fitur pencadangan atau sinkronisasi cloud di aplikasi tersebut untuk membantu Anda menghindari kehilangan akses ke akun jika Anda kehilangan atau merusak perangkat Anda dengan aplikasi autentikator.

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Pengguna Anda dapat bergabung AWS dengan penyedia identitas tempat mereka mengautentikasi dengan kredensi dan konfigurasi perusahaan mereka. MFA Untuk mengelola akses ke AWS dan aplikasi bisnis, kami sarankan Anda menggunakan Pusat IAM Identitas. Untuk informasi selengkapnya, lihat Panduan Pengguna Pusat IAM Identitas.

Topik

- Izin diperlukan
- Aktifkan kunci sandi atau kunci keamanan untuk IAM pengguna Anda sendiri (konsol)
- Aktifkan kunci sandi atau kunci keamanan untuk IAM pengguna lain (konsol)
- Ganti kunci sandi atau kunci keamanan
- Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

Izin diperlukan

Untuk mengelola FIDO kunci sandi bagi IAM pengguna Anda sendiri sambil melindungi tindakan sensitif MFA terkait, Anda harus memiliki izin dari kebijakan berikut:

> **Note**
>
> ARNNilainya adalah nilai statis dan bukan merupakan indikator protokol apa yang digunakan untuk mendaftarkan autentikator. Kami telah menghentikan U2F, jadi semua implementasi baru digunakan. WebAuthn

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
```

```
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Aktifkan kunci sandi atau kunci keamanan untuk IAM pengguna Anda sendiri (konsol)

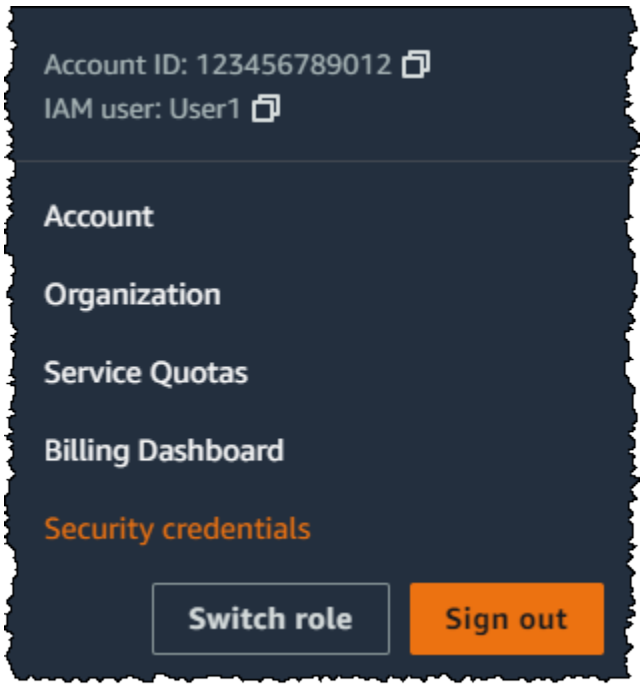
Anda dapat mengaktifkan kunci sandi atau kunci keamanan untuk IAM pengguna Anda sendiri dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API. Sebelum Anda dapat mengaktifkan kunci keamanan, Anda harus memiliki akses fisik ke perangkat.

Untuk mengaktifkan kunci sandi atau kunci keamanan untuk IAM pengguna Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama IAM pengguna, dan kata sandi Anda untuk masuk ke IAMkonsol.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama IAM pengguna dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login IAM pengguna untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.
2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Pada halaman IAM pengguna yang dipilih, pilih tab Security credentials.
4. Di bawah Autentikasi multi-faktor (MFA), pilih MFATetapkan perangkat.
5. Pada halaman nama MFA perangkat, masukkan nama Perangkat, pilih Kunci Sandi atau Kunci Keamanan, lalu pilih Berikutnya.
6. Pada Siapkan perangkat, atur kunci sandi Anda. Buat passkey dengan data biometrik seperti wajah atau sidik jari Anda, dengan pin perangkat, atau dengan memasukkan kunci FIDO keamanan ke USB port komputer Anda dan mengetuknya.
7. Ikuti petunjuk di browser Anda dan kemudian pilih Lanjutkan.

Anda sekarang telah mendaftarkan kunci sandi atau kunci keamanan Anda untuk digunakan. AWS Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihatMFAmengaktifkan login.

Aktifkan kunci sandi atau kunci keamanan untuk IAM pengguna lain (konsol)

Anda dapat mengaktifkan kunci sandi atau keamanan untuk IAM pengguna lain dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API.

Untuk mengaktifkan kunci sandi atau keamanan untuk IAM pengguna lain (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Pengguna.
3. Di bawah Pengguna, pilih nama pengguna yang ingin Anda aktifkanMFA.
4. Pada halaman IAM pengguna yang dipilih, pilih tab Security Credentials.
5. Di bawah Autentikasi multi-faktor (MFA), pilih MFATetapkan perangkat.
6. Pada halaman nama MFA perangkat, masukkan nama Perangkat, pilih Kunci Sandi atau Kunci Keamanan, lalu pilih Berikutnya.
7. Pada Siapkan perangkat, atur kunci sandi Anda. Buat passkey dengan data biometrik seperti wajah atau sidik jari Anda, dengan pin perangkat, atau dengan memasukkan kunci FIDO keamanan ke USB port komputer Anda dan mengetuknya.
8. Ikuti petunjuk di browser Anda dan kemudian pilih Lanjutkan.

Anda sekarang telah mendaftarkan kunci sandi atau kunci keamanan untuk digunakan IAM pengguna lain. AWS Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihatMFAmengaktifkan login.

Ganti kunci sandi atau kunci keamanan

Anda dapat memiliki hingga delapan MFA perangkat dari kombinasi MFAjenis yang saat ini didukung yang ditetapkan untuk pengguna pada satu waktu dengan Anda Pengguna root akun AWS dan IAM pengguna. Jika pengguna kehilangan FIDO autentikator atau perlu menggantinya karena alasan apa pun, Anda harus menonaktifkan autentikator lama FIDO terlebih dahulu. Kemudian Anda dapat menambahkan MFA perangkat baru untuk pengguna.

- Untuk menonaktifkan perangkat yang saat ini dikaitkan dengan IAM pengguna, lihatNonaktifkan perangkat MFA.
- Untuk menambahkan kunci FIDO keamanan baru bagi IAM pengguna, lihatAktifkan kunci sandi atau kunci keamanan untuk IAM pengguna Anda sendiri (konsol).

Jika Anda tidak memiliki akses ke kunci sandi atau kunci keamanan baru, Anda dapat mengaktifkan MFA perangkat virtual atau TOTP token perangkat keras baru. Lihat salah satu dari yang berikut untuk petunjuk:

- Tetapkan MFA perangkat virtual di AWS Management Console
- Tetapkan TOTP token perangkat keras di AWS Management Console

Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

Anda dapat menggunakan kunci sandi FIDO2 terikat perangkat, juga dikenal sebagai kunci keamanan, sebagai metode otentikasi multi-faktor (MFA) dengan menggunakan konfigurasi yang didukung saat ini. IAM Ini termasuk FIDO2 perangkat yang didukung oleh IAM dan browser yang mendukungFIDO2. Sebelum Anda mendaftarkan FIDO2 perangkat Anda, periksa apakah Anda menggunakan versi browser dan sistem operasi (OS) terbaru. Fitur dapat berperilaku berbeda di berbagai browser, autentikator, dan klien OS. Jika pendaftaran perangkat Anda gagal pada satu browser, Anda dapat mencoba mendaftar dengan browser lain.

FIDO2adalah standar otentikasi terbuka dan perpanjangan dari FIDO U2F, menawarkan tingkat keamanan yang sama tinggi berdasarkan kriptografi kunci publik. FIDO2terdiri dari spesifikasi W3C Web Authentication (WebAuthn API) dan FIDO Alliance Client-to-Authenticator Protocol (CTAP), sebuah protokol lapisan aplikasi. CTAPmemungkinkan komunikasi antara klien atau platform, seperti browser atau sistem operasi, dengan autentikator eksternal. Saat Anda mengaktifkan autentikator FIDO Bersertifikat AWS, kunci keamanan akan membuat key pair baru untuk digunakan dengan saja AWS. Pertama, Anda memasukkan kredensial Anda. Saat diminta, Anda mengetuk kunci keamanan, yang merespons tantangan otentikasi yang dikeluarkan oleh. AWS Untuk mempelajari lebih lanjut tentang FIDO2 standar, lihat FIDO2Proyek.

FIDO2perangkat yang didukung oleh AWS

IAMmendukung perangkat FIDO2 keamanan yang terhubung ke perangkat Anda melaluiUSB, Bluetooth, atauNFC. IAMjuga mendukung otentikator platform seperti TouchID atau FaceID. IAMtidak mendukung pendaftaran passkey lokal untuk Windows Hello. Untuk membuat dan menggunakan kunci sandi, pengguna Windows harus menggunakan otentikasi lintas perangkat di mana Anda menggunakan kunci sandi dari satu perangkat seperti perangkat seluler atau kunci keamanan perangkat keras untuk masuk di perangkat lain seperti laptop.

> **Note**
>
> AWS memerlukan akses ke USB port fisik di komputer Anda untuk memverifikasi FIDO2 perangkat Anda. Kunci keamanan tidak akan berfungsi dengan mesin virtual, koneksi jarak jauh, atau mode penyamaran browser.

FIDOAliansi menyimpan daftar semua FIDO2produk yang kompatibel dengan FIDO spesifikasi.

Browser yang mendukung FIDO2

Ketersediaan perangkat FIDO2 keamanan yang berjalan di browser web tergantung pada kombinasi browser dan sistem operasi. Browser berikut saat ini mendukung penggunaan kunci keamanan:

Browser web	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Ya	Ya	Ya	Ya	Tidak
Safari	Ya	Tidak	Tidak	Ya	Tidak
Edge	Ya	Ya	Tidak	Ya	Tidak
Firefox	Ya	Ya	Tidak	Ya	Tidak

> **Note**
>
> Sebagian besar versi Firefox yang saat ini mendukung FIDO2 tidak mengaktifkan dukungan secara default. Untuk petunjuk tentang mengaktifkan FIDO2 dukungan di Firefox, lihatMemecahkan masalah kunci keamanan FIDO.

Untuk informasi selengkapnya tentang dukungan browser untuk perangkat FIDO2 -Certified seperti YubiKey, lihat Sistem operasi dan dukungan browser web untuk FIDO2 dan U2F.

Plugin peramban

AWS hanya mendukung browser yang mendukung FIDO2 secara native. AWS tidak mendukung penggunaan plugin untuk menambahkan dukungan FIDO2 browser. Beberapa plugin browser tidak kompatibel dengan FIDO2 standar dan dapat menyebabkan hasil yang tidak terduga dengan kunci FIDO2 keamanan.

Untuk informasi tentang menonaktifkan plugin peramban dan tips pemecahan masalah lainnya, lihat Saya tidak dapat mengaktifkan kunci FIDO keamanan saya.

Sertifikasi perangkat

Kami menangkap dan menetapkan sertifikasi terkait perangkat, seperti FIPS validasi dan tingkat FIDO sertifikasi, hanya selama pendaftaran kunci keamanan. Sertifikasi perangkat Anda diambil dari

Layanan Metadata FIDO Aliansi (). MDS Jika status sertifikasi atau tingkat kunci keamanan Anda berubah, itu tidak akan tercermin dalam tag perangkat secara otomatis. Untuk memperbarui informasi sertifikasi perangkat, daftarkan perangkat lagi untuk mengambil informasi sertifikasi yang diperbarui.

AWS menyediakan jenis sertifikasi berikut sebagai kunci kondisi selama pendaftaran perangkat, diperoleh dari FIDOMDS: FIPS -140-2, -140-3, dan FIPS tingkat sertifikasi. FIDO Anda memiliki kemampuan untuk menentukan pendaftaran autentikator tertentu dalam IAM kebijakan mereka, berdasarkan jenis dan tingkat sertifikasi pilihan Anda. Untuk informasi selengkapnya, lihat kebijakan di bawah ini.

Contoh kebijakan untuk sertifikasi perangkat

Kasus penggunaan berikut menunjukkan contoh kebijakan yang memungkinkan Anda mendaftarkan MFA perangkat dengan FIPS sertifikasi.

Topik

- Kasus penggunaan 1: Izinkan mendaftarkan hanya perangkat yang memiliki sertifikasi FIPS -140-2 L2
- Kasus penggunaan 2: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS -140-2 L2 dan L1 FIDO
- Kasus penggunaan 3: Izinkan mendaftarkan perangkat yang memiliki sertifikasi -140-2 L2 atau FIPS -140-3 L2 FIPS
- Kasus penggunaan 4: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS -140-2 L2 dan mendukung MFA jenis lain seperti otentikator virtual dan perangkat keras TOTP

Kasus penggunaan 1: Izinkan mendaftarkan hanya perangkat yang memiliki sertifikasi FIPS -140-2 L2

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]
}
```

Kasus penggunaan 2: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS -140-2 L2 dan L1 FIDO

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
```

```
}
```

Kasus penggunaan 3: Izinkan mendaftarkan perangkat yang memiliki sertifikasi -140-2 L2 atau FIPS -140-3 L2 FIPS

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
            }
        }
    ]
}
```

Kasus penggunaan 4: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS -140-2 L2 dan mendukung MFA jenis lain seperti otentikator virtual dan perangkat keras TOTP

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Activate",
                    "iam:FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "Null": {
                    "iam:RegisterSecurityKey": "true"
                }
            }
        }
    ]
}
```

AWS CLI dan AWS API

AWS mendukung penggunaan kunci sandi dan kunci keamanan hanya di. AWS Management Console Menggunakan kunci sandi dan kunci keamanan untuk MFA tidak didukung di AWS CLIdan AWS API, atau untuk akses ke operasi MFAyang dilindungi API.

Sumber daya tambahan

- Untuk informasi selengkapnya tentang penggunaan kunci sandi dan kunci keamanan AWS, lihatTetapkan kunci sandi atau kunci keamanan di AWS Management Console.
- Untuk bantuan dalam memecahkan masalah kunci sandi dan kunci keamanan, lihat. AWSMemecahkan masalah kunci keamanan FIDO
- Untuk informasi industri umum tentang FIDO2 dukungan, lihat FIDO2Proyek.

Tetapkan MFA perangkat virtual di AWS Management Console

Anda dapat menggunakan telepon atau perangkat lain sebagai perangkat otentikasi multi-faktor virtual (MFA). Untuk melakukan ini, instal aplikasi seluler yang sesuai dengan RFC6238, algoritma berbasis standar TOTP (kata sandi satu kali berbasis waktu). Aplikasi ini menghasilkan kode otentikasi enam digit. Karena mereka dapat berjalan di perangkat seluler yang tidak aman, virtual MFA mungkin tidak memberikan tingkat keamanan yang sama dengan kunci FIDO keamanan. Kami menyarankan Anda menggunakan MFA perangkat virtual sambil menunggu persetujuan pembelian perangkat keras atau saat Anda menunggu perangkat keras Anda tiba.

Sebagian besar MFA aplikasi virtual mendukung pembuatan beberapa perangkat virtual, memungkinkan Anda menggunakan aplikasi yang sama untuk beberapa Akun AWS atau pengguna. Anda dapat mendaftarkan hingga delapan MFA perangkat dari kombinasi MFAjenis apa pun dengan Anda Pengguna root akun AWS dan IAM pengguna. Anda hanya perlu satu MFA perangkat untuk masuk ke AWS Management Console atau membuat sesi melalui AWS CLI. Kami menyarankan Anda mendaftarkan beberapa MFA perangkat. Untuk aplikasi autentikator, kami juga menyarankan untuk mengaktifkan fitur pencadangan atau sinkronisasi cloud untuk membantu Anda menghindari kehilangan akses ke akun jika perangkat Anda hilang atau rusak.

AWS membutuhkan MFA aplikasi virtual yang menghasilkan enam digitOTP. Untuk daftar MFA aplikasi virtual yang dapat Anda gunakan, lihat Otentikasi Multi-Faktor.

Topik

- Izin diperlukan

- Aktifkan MFA perangkat virtual untuk IAM pengguna (konsol)
- Ganti MFA perangkat virtual

Izin diperlukan

Untuk mengelola MFA perangkat virtual bagi IAM pengguna Anda, Anda harus memiliki izin dari kebijakan berikut: AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan

Aktifkan MFA perangkat virtual untuk IAM pengguna (konsol)

Anda dapat menggunakan IAM dalam AWS Management Console untuk mengaktifkan dan mengelola MFA perangkat virtual untuk IAM pengguna di akun Anda. Anda dapat melampirkan tag ke IAM sumber daya Anda, termasuk MFA perangkat virtual, untuk mengidentifikasi, mengatur, dan mengontrol akses ke mereka. Anda dapat menandai MFA perangkat virtual hanya ketika Anda menggunakan AWS CLI atau AWS API. Untuk mengaktifkan dan mengelola MFA perangkat menggunakan AWS CLI atau AWS API, lihatTetapkan MFA perangkat di atau AWS CLIAWS API. Untuk informasi selengkapnya tentang menandai IAM sumber daya, lihatTag untuk AWS Identity and Access Management sumber daya.

> **Note**
>
> Anda harus memiliki akses fisik ke perangkat keras yang akan meng-host MFA perangkat virtual pengguna untuk mengkonfigurasiMFA. Misalnya, Anda dapat mengonfigurasi MFA untuk pengguna yang akan menggunakan MFA perangkat virtual yang berjalan di ponsel cerdas. Dalam hal ini, Anda harus memiliki smartphone yang tersedia untuk menyelesaikan wizard. Karena itu, Anda mungkin ingin membiarkan pengguna mengonfigurasi dan mengelola MFA perangkat virtual mereka sendiri. Dalam hal ini, Anda harus memberi pengguna izin untuk melakukan IAM tindakan yang diperlukan. Untuk informasi selengkapnya dan contoh IAM kebijakan yang memberikan izin ini, lihat kebijakan IAMtutorial: Izinkan pengguna untuk mengelola kredensi dan pengaturan mereka MFA dan contoh. AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan

Untuk mengaktifkan MFA perangkat virtual untuk IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar Pengguna, pilih nama IAM pengguna.
4. Pilih tab Kredensial Keamanan. Di bawah Autentikasi multi-faktor (MFA), pilih MFATetapkan perangkat.
5. Di wizard, ketikkan nama Perangkat, pilih Aplikasi Authenticator, lalu pilih Berikutnya.

   IAMmenghasilkan dan menampilkan informasi konfigurasi untuk MFA perangkat virtual, termasuk grafik kode QR. Grafik adalah representasi "kunci konfigurasi rahasia" yang tersedia untuk entri manual pada perangkat yang tidak mendukung kode QR.
6. Buka MFA aplikasi virtual Anda. Untuk daftar aplikasi yang dapat Anda gunakan untuk hosting MFA perangkat virtual, lihat Otentikasi Multi-Faktor.

   Jika MFA aplikasi virtual mendukung beberapa MFA perangkat atau akun virtual, pilih opsi untuk membuat MFA perangkat atau akun virtual baru.
7. Tentukan apakah MFA aplikasi mendukung kode QR, lalu lakukan salah satu hal berikut:

   - Dari wizard, pilih Tampilkan kode QR, lalu gunakan aplikasi untuk memindai kode QR. Ini mungkin ikon kamera atau opsi Pindai kode yang menggunakan kamera perangkat untuk memindai kode.
   - Dari wizard, pilih Tampilkan kunci rahasia, lalu ketik kunci rahasia ke dalam MFA aplikasi Anda.

   Setelah selesai, MFA perangkat virtual mulai menghasilkan kata sandi satu kali.
8. Pada halaman Siapkan perangkat, dalam kotak MFAkode 1, ketik kata sandi satu kali yang saat ini muncul di MFA perangkat virtual. Tunggu hingga 30 detik untuk membuat kata sandi sekali pakai yang baru. Kemudian ketik kata sandi satu kali kedua ke dalam kotak MFAkode 2. Pilih Tambah MFA.

   > **Important**
   >
   > Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat

berhasil dikaitkan dengan pengguna tetapi MFA perangkat tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

MFAPerangkat virtual sekarang siap digunakan AWS. Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihatMFAmengaktifkan login.

Ganti MFA perangkat virtual

Anda Pengguna root akun AWS dan IAM pengguna dapat mendaftarkan hingga delapan MFA perangkat dari kombinasi MFA jenis apa pun. Jika pengguna kehilangan perangkat atau perlu menggantinya karena alasan apa pun, nonaktifkan perangkat lama. Kemudian Anda dapat menambahkan perangkat baru untuk pengguna.

- Untuk menonaktifkan perangkat yang saat ini terkait dengan IAM pengguna lain, lihatNonaktifkan perangkat MFA.
- Untuk menambahkan MFA perangkat virtual pengganti untuk IAM pengguna lain, ikuti langkah-langkah dalam prosedur di Aktifkan MFA perangkat virtual untuk IAM pengguna (konsol) atas.
- Untuk menambahkan MFA perangkat virtual pengganti untuk Pengguna root akun AWS, ikuti langkah-langkah dalam prosedurAktifkan MFA perangkat virtual untuk pengguna root (konsol).

Tetapkan TOTP token perangkat keras di AWS Management Console

TOTPToken perangkat keras menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali () berbasis waktu. TOTP Pengguna harus memasukkan kode yang valid dari perangkat saat diminta selama proses masuk. Setiap MFA perangkat yang ditetapkan untuk pengguna harus unik; pengguna tidak dapat mengetik kode dari perangkat pengguna lain untuk diautentikasi. MFAperangkat tidak dapat dibagikan di seluruh akun atau pengguna.

TOTPToken perangkat keras dan kunci FIDO keamanan adalah perangkat fisik yang Anda beli. Perangkat MFA perangkat keras menghasilkan TOTP kode untuk otentikasi saat Anda masuk AWS. Mereka mengandalkan baterai, yang mungkin perlu penggantian dan sinkronisasi ulang AWS seiring waktu. FIDOkunci keamanan, yang menggunakan kriptografi kunci publik, tidak memerlukan baterai dan menawarkan proses otentikasi yang mulus. Sebaiknya gunakan kunci FIDO keamanan untuk ketahanan phishing mereka, yang memberikan alternatif yang lebih aman untuk TOTP perangkat. Selain itu, kunci FIDO keamanan dapat mendukung beberapa IAM atau pengguna root pada

perangkat yang sama, meningkatkan utilitas mereka untuk keamanan akun. Untuk spesifikasi dan informasi pembelian untuk kedua jenis perangkat, lihat Autentikasi Multi-Faktor.

Anda dapat mengaktifkan TOTP token perangkat keras untuk IAM pengguna dari AWS Management Console, baris perintah, atau file IAMAPI. Untuk mengaktifkan MFA perangkat untuk Anda Pengguna root akun AWS, lihatAktifkan TOTP token perangkat keras untuk pengguna root (konsol).

Anda dapat mendaftarkan hingga delapan MFA perangkat dari kombinasi MFAjenis yang saat ini didukung dengan Anda Pengguna root akun AWS dan IAM pengguna. Dengan beberapa MFA perangkat, Anda hanya perlu satu MFA perangkat untuk masuk ke AWS Management Console atau membuat sesi melalui AWS CLI sebagai pengguna tersebut.

> **Important**
>
> Kami menyarankan Anda mengaktifkan beberapa MFA perangkat untuk pengguna Anda untuk melanjutkan akses ke akun Anda jika MFA perangkat hilang atau tidak dapat diakses.

> **Note**
>
> Jika Anda ingin mengaktifkan MFA perangkat dari baris perintah, gunakan `aws iam enable-mfa-device`. Untuk mengaktifkan MFA perangkat dengan IAMAPI, gunakan `EnableMFADevice`operasi.

Topik

- Izin diperlukan
- Aktifkan TOTP token perangkat keras untuk IAM pengguna Anda sendiri (konsol)
- Aktifkan TOTP token perangkat keras untuk IAM pengguna lain (konsol)
- Ganti MFA perangkat fisik

Izin diperlukan

Untuk mengelola TOTP token perangkat keras untuk IAM pengguna Anda sendiri sambil melindungi tindakan sensitif MFA terkait, Anda harus memiliki izin dari kebijakan berikut:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Aktifkan TOTP token perangkat keras untuk IAM pengguna Anda sendiri (konsol)

Anda dapat mengaktifkan TOTP token perangkat keras Anda sendiri dari file AWS Management Console.

> **Note**
>
> Sebelum Anda dapat mengaktifkan TOTP token perangkat keras, Anda harus memiliki akses fisik ke perangkat.

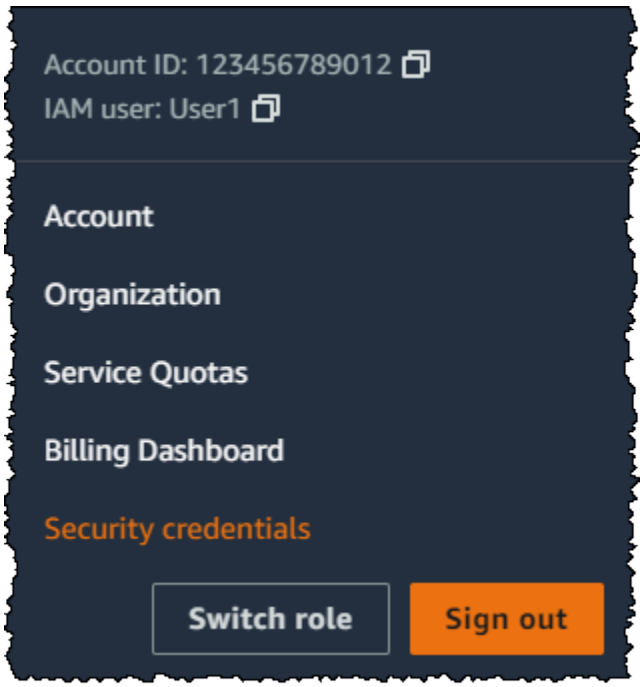
Untuk mengaktifkan TOTP token perangkat keras untuk IAM pengguna Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama IAM pengguna, dan kata sandi Anda untuk masuk ke IAMkonsol.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama IAM pengguna dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login IAM pengguna untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Pada tab AWS IAMkredensial, di bagian Autentikasi multi-faktor (MFA), pilih Tetapkan perangkat. MFA
4. Di wizard, ketikkan nama Perangkat, pilih TOTPToken perangkat keras, lalu pilih Berikutnya.
5. Ketik nomor seri perangkat. Nomor seri biasanya ada di bagian belakang perangkat.

6. Dalam kotak MFAkode 1, ketik angka enam digit yang ditampilkan oleh perangkat. MFA Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor.

7. Tunggu 30 detik saat perangkat menyegarkan kode, lalu ketik angka enam digit berikutnya ke dalam kotak kode 2. MFA Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor kedua.
8. Pilih Tambah MFA.

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat berhasil dikaitkan dengan pengguna tetapi MFA perangkat menjadi tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat siap digunakan dengan AWS. Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihatMFAmengaktifkan login.

Aktifkan TOTP token perangkat keras untuk IAM pengguna lain (konsol)

Anda dapat mengaktifkan TOTP token perangkat keras untuk IAM pengguna lain dari file AWS Management Console.

Untuk mengaktifkan TOTP token perangkat keras untuk IAM pengguna lain (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang ingin Anda aktifkanMFA.
4. Pilih tab Kredensial Keamanan. Di bawah Autentikasi multi-faktor (MFA), pilih MFATetapkan perangkat.
5. Di wizard, ketikkan nama Perangkat, pilih TOTPToken perangkat keras, lalu pilih Berikutnya.

6. Ketik nomor seri perangkat. Nomor seri biasanya ada di bagian belakang perangkat.
7. Dalam kotak MFAkode 1, ketik angka enam digit yang ditampilkan oleh perangkat. MFA Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor.

8. Tunggu 30 detik saat perangkat menyegarkan kode, lalu ketik angka enam digit berikutnya ke dalam kotak kode 2. MFA Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor kedua.
9. Pilih Tambah MFA.

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat berhasil dikaitkan dengan pengguna tetapi MFA perangkat menjadi tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat siap digunakan dengan AWS. Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihatMFAmengaktifkan login.

Ganti MFA perangkat fisik

Anda dapat memiliki hingga delapan MFA perangkat dari kombinasi MFAjenis yang saat ini didukung yang ditetapkan untuk pengguna pada satu waktu dengan Anda Pengguna root akun AWS dan IAM pengguna. Jika pengguna kehilangan perangkat atau perlu mengganti karena alasan apa pun, Anda harus menonaktifkan perangkat lama terlebih dahulu. Kemudian Anda dapat menambahkan perangkat baru untuk pengguna.

- Untuk menonaktifkan perangkat yang saat ini terkait dengan pengguna, lihat Nonaktifkan perangkat MFA.
- Untuk menambahkan TOTP token perangkat keras pengganti bagi IAM pengguna, ikuti langkah-langkah dalam prosedur Aktifkan TOTP token perangkat keras untuk IAM pengguna lain (konsol) sebelumnya dalam topik ini.

- Untuk menambahkan TOTP token perangkat keras pengganti Pengguna root akun AWS, ikuti langkah-langkah dalam prosedur Aktifkan TOTP token perangkat keras untuk pengguna root (konsol) sebelumnya dalam topik ini.

Tetapkan MFA perangkat di atau AWS CLIAWS API

Anda dapat menggunakan AWS CLI perintah atau AWS API operasi untuk mengaktifkan MFA perangkat virtual bagi IAM pengguna. Anda tidak dapat mengaktifkan MFA perangkat untuk Pengguna root akun AWS dengan AWS CLI, AWS API, Alat untuk Windows PowerShell, atau alat baris perintah lainnya. Namun, Anda dapat menggunakan AWS Management Console untuk mengaktifkan MFA perangkat untuk pengguna root.

Saat Anda mengaktifkan MFA perangkat dari AWS Management Console, konsol melakukan beberapa langkah untuk Anda. Jika Anda malah membuat perangkat virtual menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API, maka Anda harus melakukan langkah-langkah secara manual dan dalam urutan yang benar. Misalnya, untuk membuat MFA perangkat virtual, Anda harus membuat IAM objek dan mengekstrak kode sebagai string atau grafik kode QR. Maka Anda harus menyinkronkan perangkat dan mengaitkannya dengan IAM pengguna. Lihat bagian Contoh Baru- IAMVirtualMFADevice untuk lebih jelasnya. Untuk perangkat fisik, Anda melompati langkah pembuatan dan langsung menuju sinkronisasi perangkat dan mengaitkannya dengan pengguna.

Anda dapat melampirkan tag ke IAM sumber daya Anda, termasuk MFA perangkat virtual, untuk mengidentifikasi, mengatur, dan mengontrol akses ke mereka. Anda dapat menandai MFA perangkat virtual hanya ketika Anda menggunakan AWS CLI atau AWS API.

IAMPengguna yang menggunakan SDK atau CLI dapat mengaktifkan MFA perangkat tambahan dengan memanggil `EnableMFADevice`atau menonaktifkan MFA perangkat yang ada dengan menelepon `DeactivateMFADevice`. Untuk melakukan ini dengan sukses, mereka harus terlebih dahulu memanggil `GetSessionToken`dan mengirimkan MFA kode dengan MFA perangkat yang ada. Panggilan ini mengembalikan kredensi keamanan sementara yang kemudian dapat digunakan untuk menandatangani API operasi yang memerlukan MFA otentikasi. Untuk contoh permintaan dan respons, lihat `GetSessionToken`—kredensial sementara untuk pengguna di lingkungan yang tidak tepercaya.

Untuk membuat entitas perangkat virtual IAM untuk mewakili MFA perangkat virtual

Perintah ini menyediakan ARN untuk perangkat yang digunakan sebagai pengganti nomor seri di banyak perintah berikut.

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API: `CreateVirtualMFADevice`

Untuk mengaktifkan MFA perangkat untuk digunakan dengan AWS

Perintah ini menyinkronkan perangkat dengan AWS dan mengaitkannya dengan pengguna. Jika perangkat virtual, gunakan perangkat virtual sebagai nomor seri. ARN

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat berhasil dikaitkan dengan pengguna tetapi MFA perangkat menjadi tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat mensinkronisasi ulang perangkat menggunakan perintah yang dijelaskan di bawah ini.

- AWS CLI: `aws iam enable-mfa-device`
- AWS API: `EnableMFADevice`

Untuk menonaktifkan perangkat

Gunakan perintah ini untuk memisahkan perangkat dari pengguna dan menonaktifkannya. Jika perangkat virtual, gunakan perangkat virtual sebagai nomor seri. ARN Anda juga harus menghapus perangkat virtual secara terpisah.

- AWS CLI: `aws iam deactivate-mfa-device`
- AWS API: `DeactivateMFADevice`

Untuk mencantumkan entitas MFA perangkat virtual

Gunakan perintah ini untuk membuat daftar entitas MFA perangkat virtual.

- AWS CLI: `aws iam list-virtual-mfa-devices`
- AWS API: `ListVirtualMFADevices`

Untuk menandai MFA perangkat virtual

Gunakan perintah ini untuk menandai MFA perangkat virtual.

- AWS CLI: `aws iam tag-mfa-device`
- AWS API: `TagMFADevice`

Untuk membuat daftar tag untuk MFA perangkat virtual

Gunakan perintah ini untuk membuat daftar tag yang dilampirkan ke MFA perangkat virtual.

- AWS CLI: `aws iam list-mfa-device-tags`
- AWS API: `ListMFADeviceTags`

Untuk menghapus tag perangkat virtual MFA

Gunakan perintah ini untuk menghapus tag yang dilampirkan ke MFA perangkat virtual.

- AWS CLI: `aws iam untag-mfa-device`
- AWS API: `UntagMFADevice`

Untuk menyinkronkan ulang perangkat MFA

Gunakan perintah ini jika perangkat menghasilkan kode yang tidak diterima oleh AWS. Jika perangkat virtual, gunakan perangkat virtual sebagai nomor seri. ARN

- AWS CLI: `aws iam resync-mfa-device`
- AWS API: `ResyncMFADevice`

Untuk menghapus entitas MFA perangkat virtual di IAM

Setelah perangkat diputus kaitannya dari pengguna, Anda dapat menghapus entitas perangkat.

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API: `DeleteVirtualMFADevice`

Untuk memulihkan MFA perangkat virtual yang hilang atau tidak berfungsi

Terkadang, perangkat pengguna yang meng-host MFA aplikasi virtual hilang, diganti, atau tidak berfungsi. Jika ini terjadi, pengguna tidak dapat memulihkannya sendiri. Pengguna harus menghubungi administrator untuk menonaktifkan perangkat. Untuk informasi selengkapnya, lihat Memulihkan identitas yang MFA dilindungi di IAM.

Periksa MFA status

Gunakan IAM konsol untuk memeriksa apakah IAM pengguna Pengguna root akun AWS atau memiliki MFA perangkat yang valid diaktifkan.

Untuk memeriksa MFA status pengguna root

1. Masuk ke AWS Management Console dengan kredensi pengguna root Anda dan kemudian buka IAM konsol di. https://console.aws.amazon.com/iam/
2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.
3. Periksa di bawah Multi-faktor Authentication (MFA) untuk melihat apakah MFA diaktifkan atau dinonaktifkan. Jika MFA belum diaktifkan, simbol peringatan ( ) ditampilkan.

Jika Anda ingin mengaktifkan MFA akun, lihat salah satu dari berikut ini:

- Aktifkan MFA perangkat virtual untuk pengguna root (konsol)
- Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)
- Aktifkan TOTP token perangkat keras untuk pengguna root (konsol)

Untuk memeriksa MFA status IAM pengguna

1. Membuka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan MFAkolom ke tabel pengguna dengan menyelesaikan langkah-langkah berikut:
   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ( ).
   b. Di Kelola Kolom, pilih MFA.

c. (Opsional) Kosongkan kotak centang untuk judul kolom yang tidak ingin Anda tampilkan dalam tabel pengguna.

d. Pilih Tutup untuk kembali ke daftar pengguna.

4. MFAKolom memberi tahu Anda tentang MFA perangkat yang diaktifkan. Jika tidak ada MFA perangkat yang aktif untuk pengguna, konsol akan menampilkan None. Jika pengguna mengaktifkan MFA perangkat, MFAkolom akan menampilkan jenis perangkat yang diaktifkan dengan nilai Virtual, Kunci Keamanan, Perangkat Keras, atau SMS.

> **Note**
>
> AWS mengakhiri dukungan untuk mengaktifkan otentikasi SMS multi-faktor (). MFA Kami menyarankan agar pelanggan yang memiliki IAM pengguna yang menggunakan pesan SMS teks berbasis MFA beralih ke salah satu metode alternatif berikut: perangkat virtual (berbasis perangkat lunak), kunci FIDO keamanan, atau MFA perangkat keras. MFA Anda dapat mengidentifikasi pengguna di akun Anda dengan SMS MFA perangkat yang ditetapkan. Untuk melakukannya, buka IAM konsol, pilih Pengguna dari panel navigasi, dan cari pengguna dengan SMSdi MFAkolom tabel.

5. Untuk melihat informasi tambahan tentang MFA perangkat bagi pengguna, pilih nama pengguna yang MFA statusnya ingin Anda periksa. Kemudian pilih tab Kredensial keamanan.

6. Jika tidak ada MFA perangkat yang aktif untuk pengguna, konsol akan menampilkan Tidak ada MFA perangkat. Tetapkan MFA perangkat untuk meningkatkan keamanan AWS lingkungan Anda di bagian Autentikasi multi-faktor () MFA. Jika pengguna mengaktifkan MFA perangkat, bagian Autentikasi multi-faktor (MFA) menampilkan detail tentang perangkat:

   - Nama perangkat
   - Jenis perangkat
   - Pengenal untuk perangkat, seperti nomor seri untuk perangkat fisik atau ARN in AWS untuk perangkat virtual
   - Saat perangkat dibuat

Untuk menghapus atau menyinkronkan ulang perangkat, pilih tombol radio di sebelah perangkat dan pilih Hapus atau Sinkronkan Ulang.

Untuk informasi selengkapnya tentang mengaktifkanMFA, lihat berikut ini:

- Tetapkan MFA perangkat virtual di AWS Management Console
- Tetapkan kunci sandi atau kunci keamanan di AWS Management Console
- Tetapkan TOTP token perangkat keras di AWS Management Console

Sinkronisasi ulang perangkat virtual dan perangkat keras MFA

Anda dapat menggunakan AWS untuk menyinkronkan ulang perangkat otentikasi multi-faktor virtual dan perangkat keras Anda. MFA Jika perangkat Anda tidak disinkronkan saat Anda mencoba menggunakannya, upaya masuk gagal dan IAM meminta Anda untuk menyinkronkan ulang perangkat.

> **Note**
>
> FIDOkunci keamanan tidak keluar dari sinkron. Jika kunci FIDO keamanan hilang atau rusak, Anda dapat menonaktifkannya. Untuk petunjuk tentang menonaktifkan semua jenis MFA perangkat, lihat. Untuk menonaktifkan MFA perangkat untuk IAM pengguna lain (konsol)

Sebagai AWS administrator, Anda dapat menyinkronkan ulang perangkat virtual dan MFA perangkat keras IAM pengguna Anda jika mereka keluar dari sinkronisasi.

Jika Pengguna root akun AWS MFA perangkat tidak berfungsi, Anda dapat menyinkronkan ulang perangkat menggunakan IAM konsol dengan atau tanpa menyelesaikan proses masuk. Jika Anda tidak berhasil menyinkronkan ulang perangkat Anda, Anda mungkin perlu menghapus kaitannya dan mengaitkannya kembali. Untuk informasi lebih lanjut tentang cara melakukan ini, lihat Nonaktifkan perangkat MFA danAWS Otentikasi multi-faktor di IAM.

Topik

- Izin diperlukan
- Menyinkronkan ulang perangkat virtual dan MFA perangkat keras (konsol) IAM
- Menyinkronkan ulang perangkat virtual dan MFA perangkat keras ()AWS CLI
- Menyinkronkan ulang perangkat virtual dan MFA perangkat keras ()AWS API

Izin diperlukan

Untuk menyinkronkan ulang perangkat virtual atau MFA perangkat keras untuk IAM pengguna Anda sendiri, Anda harus memiliki izin dari kebijakan berikut. Kebijakan ini tidak mengizinkan Anda untuk membuat atau menonaktifkan perangkat.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Menyinkronkan ulang perangkat virtual dan MFA perangkat keras (konsol) IAM

Anda dapat menggunakan IAM konsol untuk menyinkronkan ulang perangkat virtual dan perangkat kerasMFA.

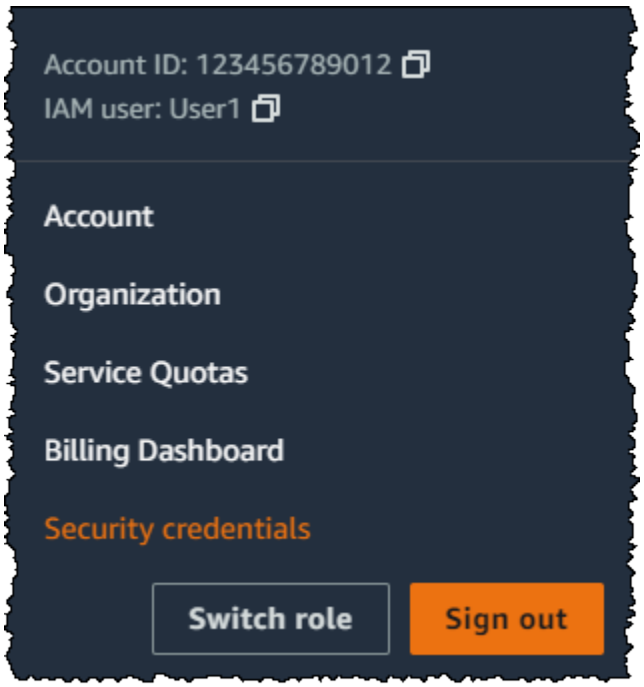
Untuk menyinkronkan ulang perangkat virtual atau MFA perangkat keras untuk IAM pengguna Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama IAM pengguna, dan kata sandi Anda untuk masuk ke IAMkonsol.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama IAM pengguna dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login IAM pengguna untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Pada tab AWS IAMkredensial, di bagian Autentikasi multi-faktor (MFA), pilih tombol radio di sebelah MFA perangkat dan pilih Resinkronisasi.
4. Ketik dua kode berikutnya yang dihasilkan secara berurutan dari perangkat ke dalam MFAkode 1 dan MFAkode 2. Kemudian pilih Resinkronisasi.

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggunya terlalu lama untuk mengirimkan permintaan, permintaan tersebut muncul untuk bekerja tetapi perangkat tetap tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat.

Untuk menyinkronkan ulang perangkat virtual atau MFA perangkat keras untuk IAM pengguna lain (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna, lalu pilih nama pengguna yang MFA perangkatnya perlu disinkronkan ulang.
3. Pilih tab Kredensial keamanan. Di bagian Multi-factor authentication (MFA), pilih tombol radio di sebelah MFA perangkat dan pilih Ressync.
4. Ketik dua kode berikutnya yang dihasilkan secara berurutan dari perangkat ke dalam MFAkode 1 dan MFAkode 2. Kemudian pilih Resinkronisasi.

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggunya terlalu lama untuk mengirimkan permintaan, permintaan tersebut muncul untuk bekerja tetapi perangkat tetap tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat.

Untuk menyinkronkan ulang pengguna root Anda MFA sebelum masuk (konsol)

1. Di halaman Masuk Amazon Web Services Dengan Perangkat Autentikasi, pilih Ada masalah dengan perangkat autentikasi Anda? Klik di sini.

> **Note**
>
> Anda mungkin melihat teks yang berbeda, seperti Masuk menggunakan MFA dan Memecahkan masalah perangkat autentikasi Anda. Namun, fitur yang sama disediakan.

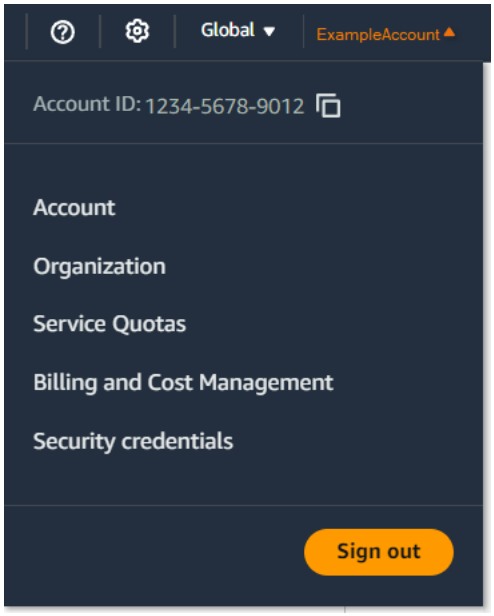
2. Di bagian Re-Sync With Our Server, ketikkan dua kode berikutnya yang dihasilkan secara berurutan dari perangkat ke dalam MFAkode 1 dan MFA kode 2. Lalu, pilih Sinkronkan ulang perangkat autentikasi.
3. Jika perlu, ketikkan kata sandi Anda kembali dan pilih Masuk. Kemudian selesaikan proses masuk menggunakan MFA perangkat Anda.

Untuk menyinkronkan ulang MFA perangkat pengguna root Anda setelah masuk (konsol)

1. Masuk ke IAMkonsol sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

> **Note**
>
> Sebagai pengguna root, Anda tidak dapat masuk ke halaman Masuk sebagai IAM pengguna. Jika Anda melihat halaman Masuk sebagai IAM pengguna, pilih Masuk menggunakan email pengguna root di dekat bagian bawah halaman. Untuk bantuan masuk sebagai pengguna root, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.

2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi keamanan. Jika perlu, pilih Continue to Security credentials.



3. Perluas bagian Autentikasi Multi-faktor (MFA) pada halaman.
4. Pilih tombol radio di sebelah perangkat dan pilih Resinkronisasi.
5. Di kotak dialog MFAperangkat Resinkronisasi, ketik dua kode berikutnya yang dihasilkan secara berurutan dari perangkat ke dalam MFAkode 1 dan MFA kode 2. Kemudian pilih Resinkronisasi.

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, MFA perangkat berhasil dikaitkan dengan pengguna, tetapi MFA perangkat tidak sinkron. Ini terjadi karena kata sandi satu kali berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat.

Menyinkronkan ulang perangkat virtual dan MFA perangkat keras ()AWS CLI

Anda dapat menyinkronkan ulang perangkat virtual dan MFA perangkat keras dari file. AWS CLI

Untuk menyinkronkan ulang perangkat virtual atau MFA perangkat keras untuk IAM pengguna ()AWS CLI

Pada prompt perintah, keluarkan perintah aws iam: resync-mfa-device

- MFAPerangkat virtual: Tentukan Nama Sumber Daya Amazon (ARN) perangkat sebagai nomor seri.

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- Perangkat MFA perangkat keras: Tentukan nomor seri perangkat keras sebagai nomor seri. Formatnya khusus vendor. Misalnya, Anda dapat membeli token gemalto dari Amazon. Nomor serinya biasa terdiri atas empat huruf, diikuti empat angka.

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggunya terlalu lama untuk mengirimkan permintaan, permintaan gagal karena kode kedaluwarsa setelah beberapa saat.

Menyinkronkan ulang perangkat virtual dan MFA perangkat keras ()AWS API

IAMmemiliki API panggilan yang melakukan sinkronisasi. Dalam hal ini, kami menyarankan Anda memberikan izin kepada pengguna MFA perangkat virtual dan perangkat keras Anda untuk mengakses API panggilan ini. Kemudian buat alat berdasarkan API panggilan itu sehingga pengguna Anda dapat menyinkronkan ulang perangkat mereka kapan pun mereka membutuhkannya.

Untuk menyinkronkan ulang perangkat virtual atau MFA perangkat keras untuk IAM pengguna ()AWS API

- Kirim esyncMFADevice permintaan R.

Nonaktifkan perangkat MFA

Jika Anda mengalami masalah saat masuk dengan perangkat autentikasi multi-faktor (MFA) sebagai IAM pengguna, hubungi administrator untuk mendapatkan bantuan.

Sebagai administrator, Anda dapat menonaktifkan perangkat untuk IAM pengguna lain. Ini memungkinkan pengguna untuk masuk tanpa menggunakanMFA. Anda dapat melakukan ini sebagai

solusi sementara saat MFA perangkat diganti, atau jika perangkat sementara tidak tersedia. Namun, kami menyarankan Anda untuk mengaktifkan perangkat baru untuk pengguna sesegera mungkin. Untuk mempelajari cara mengaktifkan MFA perangkat baru, lihatAWS Otentikasi multi-faktor di IAM.

> **Note**
>
> Jika Anda menggunakan API atau AWS CLI untuk menghapus pengguna dari Anda Akun AWS, Anda harus menonaktifkan atau menghapus MFA perangkat pengguna. Anda membuat perubahan ini sebagai bagian dari proses penghapusan pengguna. Untuk informasi lebih lanjut tentang penghapusan pengguna, lihat Menghapus atau menonaktifkan pengguna IAM.

Topik

- Menonaktifkan MFA perangkat (konsol)
- Menonaktifkan MFA perangkat ()AWS CLI
- Menonaktifkan MFA perangkat ()AWS API

Menonaktifkan MFA perangkat (konsol)

Untuk menonaktifkan MFA perangkat untuk IAM pengguna lain (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Untuk menonaktifkan MFA perangkat bagi pengguna, pilih nama pengguna yang ingin MFA Anda hapus.
4. Pilih tab Kredensial keamanan.
5. Di bawah Autentikasi multi-faktor (MFA), pilih tombol radio di sebelah MFA perangkat, pilih Hapus, lalu pilih Hapus.

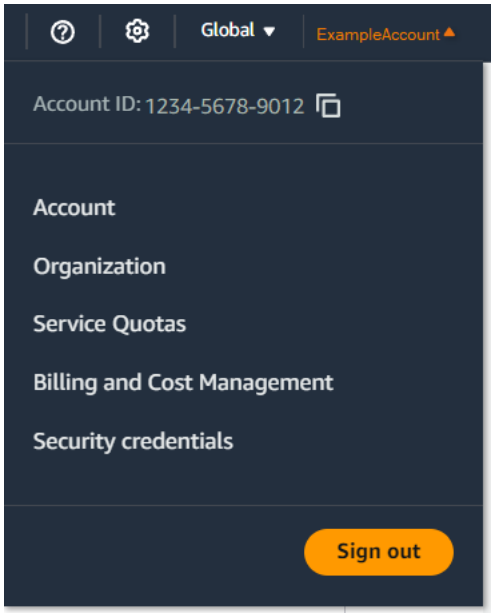
   Perangkat dihapus dari AWS. Ini tidak dapat digunakan untuk masuk atau mengotentikasi permintaan sampai diaktifkan kembali dan dikaitkan dengan AWS pengguna atau. Pengguna root akun AWS

Untuk menonaktifkan MFA perangkat untuk Pengguna root akun AWS (konsol) Anda

1. Masuk ke IAMkonsol sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   > **Note**
   >
   > Sebagai pengguna root, Anda tidak dapat masuk ke halaman Masuk sebagai IAM pengguna. Jika Anda melihat halaman Masuk sebagai IAM pengguna, pilih Masuk menggunakan email pengguna root di dekat bagian bawah halaman. Untuk bantuan masuk sebagai pengguna root, lihat Masuk ke AWS Management Console sebagai pengguna root di Panduan AWS Sign-In Pengguna.

2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi keamanan. Jika perlu, pilih Continue to Security credentials.



3. Di bagian Autentikasi multi-faktor (MFA), pilih tombol radio di sebelah MFA perangkat yang ingin Anda nonaktifkan dan pilih Hapus.
4. Pilih Hapus.

   MFAPerangkat dinonaktifkan untuk file. Akun AWS Periksa email yang terkait dengan Anda Akun AWS untuk pesan konfirmasi dari Amazon Web Services. Email tersebut memberi tahu Anda bahwa autentikasi multi-faktor Amazon Web Services (MFA) Anda telah dinonaktifkan. Pesan akan datang dari `@amazon.com` atau`@aws.amazon.com`.

Menonaktifkan MFA perangkat ()AWS CLI

Untuk menonaktifkan MFA perangkat untuk IAM pengguna ()AWS CLI

- Jalankan perintah ini: `aws iam deactivate-mfa-device`

Menonaktifkan MFA perangkat ()AWS API

Untuk menonaktifkan MFA perangkat untuk IAM pengguna ()AWS API

- Hubungi operasi ini: `DeactivateMFADevice`

Memulihkan identitas yang MFA dilindungi di IAM

Jika MFAperangkat virtual atau TOTPtoken perangkat keras Anda tampaknya berfungsi dengan baik, tetapi Anda tidak dapat menggunakannya untuk mengakses AWS sumber daya Anda, itu mungkin tidak sinkronisasi dengan AWS. Untuk informasi tentang menyinkronkan MFA perangkat virtual atau MFA perangkat keras, lihatSinkronisasi ulang perangkat virtual dan perangkat keras MFA. FIDOkunci keamanan tidak keluar dari sinkron.

Jika MFAperangkat untuk a Pengguna root akun AWS hilang, rusak, atau tidak berfungsi, Anda dapat memulihkan akses ke akun Anda. IAMpengguna harus menghubungi administrator untuk menonaktifkan perangkat.

> **Important**
>
> Kami menyarankan Anda mengaktifkan beberapa MFA perangkat. Mendaftarkan beberapa MFA perangkat membantu memastikan akses berkelanjutan jika perangkat hilang atau rusak. Anda Pengguna root akun AWS dan IAM pengguna dapat mendaftarkan hingga delapan MFA perangkat jenis apa pun.

Prasyarat - Gunakan perangkat lain MFA

Jika perangkat otentikasi multi-faktor Anda hilang, rusak, atau tidak berfungsi, Anda dapat masuk menggunakan MFA perangkat lain yang terdaftar ke pengguna atau IAM pengguna root yang sama. MFA

Untuk masuk menggunakan MFA perangkat lain

1. Masuk ke AWS Management Consoledengan Akun AWS ID atau alias akun dan kata sandi Anda.
2. Pada halaman yang diperlukan verifikasi tambahan atau halaman otentikasi multi-faktor, pilih Coba metode lain MFA.
3. Otentikasi dengan jenis MFA perangkat yang Anda pilih.
4. Langkah selanjutnya bervariasi berdasarkan apakah Anda berhasil masuk dengan MFA perangkat alternatif.
   - Jika Anda telah berhasil masuk, Anda bisaSinkronisasi ulang perangkat virtual dan perangkat keras MFA, yang dapat menyelesaikan masalah. Jika MFA perangkat Anda hilang atau rusak, Anda dapat menonaktifkannya. Untuk petunjuk tentang menonaktifkan semua jenis MFA perangkat, lihat. Nonaktifkan perangkat MFA
   - Jika Anda tidak dapat masukMFA, gunakan langkah-langkah Memulihkan perangkat pengguna MFA root atau Memulihkan perangkat IAM pengguna MFA untuk memulihkan identitas Anda yang MFA dilindungi.

Memulihkan perangkat pengguna MFA root

Jika Anda tidak dapat masukMFA, Anda dapat menggunakan metode otentikasi alternatif untuk masuk dengan memverifikasi identitas Anda menggunakan email dan nomor telepon kontak utama yang terdaftar di akun Anda.

Konfirmasikan bahwa Anda dapat mengakses email dan nomor telepon kontak utama yang terkait dengan akun Anda sebelum Anda menggunakan faktor otentikasi alternatif untuk masuk sebagai pengguna root. Jika Anda perlu memperbarui nomor telepon kontak utama, masuk sebagai IAM pengguna dengan akses Administrator, bukan pengguna root. Untuk petunjuk tambahan tentang memperbarui informasi kontak akun, lihat Mengedit informasi kontak di Panduan AWS Billing Pengguna. Jika Anda tidak memiliki akses ke email dan nomor telepon kontak utama, Anda harus menghubungi AWS Support.

> **Important**
>
> Kami menyarankan agar Anda tetap memperbarui alamat email dan nomor telepon kontak ke pengguna root Anda agar pemulihan akun berhasil. Untuk informasi selengkapnya, lihat

Memperbarui kontak utama untuk Anda Akun AWS di Panduan AWS Account Management Referensi.

Untuk masuk menggunakan faktor alternatif otentikasi sebagai Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.
2. Pada halaman Verifikasi tambahan yang diperlukan, pilih MFA metode untuk mengautentikasi dan pilih Berikutnya.

   > **Note**
   >
   > Anda mungkin melihat teks alternatif, seperti Masuk menggunakan MFA, Memecahkan masalah perangkat autentikasi, atau Memecahkan Masalah MFA, tetapi fungsinya sama. Jika Anda tidak dapat menggunakan faktor otentikasi alternatif untuk memverifikasi alamat email akun dan nomor telepon kontak utama, hubungi AWS Supportuntuk menonaktifkan perangkat AndaMFA.

3. Tergantung pada jenis yang MFA Anda gunakan, Anda akan melihat halaman yang berbeda, tetapi MFA opsi Pemecahan Masalah berfungsi sama. Pada halaman yang diperlukan verifikasi tambahan atau halaman otentikasi multi-faktor, pilih Troubleshoot. MFA
4. Jika perlu, ketikkan kata sandi Anda kembali dan pilih Masuk.
5. Pada halaman Memecahkan masalah perangkat autentikasi Anda, di bagian Masuk menggunakan faktor alternatif autentikasi, pilih Masuk menggunakan faktor alternatif.
6. Pada halaman Masuk menggunakan faktor alternatif otentikasi, autentikasi akun Anda dengan memverifikasi alamat email, pilih Kirim email verifikasi.
7. Periksa email yang terkait dengan Anda Akun AWS untuk pesan dari Amazon Web Services (recover-mfa-no-reply@verify .signin.aws). Ikuti petunjuk di dalam email.

   Jika Anda tidak melihat email di akun Anda, periksa folder spam, atau kembali ke peramban Anda dan pilih Kirim ulang email.
8. Setelah Anda memverifikasi alamat email, Anda dapat melanjutkan mengautentikasi akun Anda. Untuk memverifikasi nomor telepon kontak utama Anda, pilih Hubungi saya sekarang.
9. Jawab panggilan dari AWS dan, ketika diminta, masukkan nomor 6 digit dari AWS situs web di keypad ponsel Anda.

Jika Anda tidak menerima panggilan dari AWS, pilih Masuk untuk masuk ke konsol lagi dan mulai dari awal. Atau lihat Perangkat Otentikasi Multi-Faktor (MFA) yang hilang atau tidak dapat digunakan untuk menghubungi dukungan untuk mendapatkan bantuan.

10. Setelah memverifikasi nomor telepon, Anda dapat masuk ke akun dengan memilih Masuk ke konsol.
11. Langkah selanjutnya bervariasi tergantung pada jenis yang MFA Anda gunakan:

    - Untuk MFA perangkat virtual, hapus akun dari perangkat Anda. Lalu pergi ke halaman AWS Security Credentials dan hapus entitas perangkat MFA virtual lama sebelum Anda membuat yang baru.
    - Untuk kunci FIDO keamanan, buka halaman AWS Security Credentials dan nonaktifkan kunci FIDO keamanan lama sebelum mengaktifkan yang baru.
    - Untuk TOTP token perangkat keras, hubungi penyedia pihak ketiga untuk bantuan memperbaiki atau mengganti perangkat. Anda dapat terus masuk menggunakan faktor alternatif autentikasi hingga Anda menerima perangkat baru. Setelah Anda memiliki MFA perangkat keras baru, buka halaman AWS Security Credentials dan hapus perangkat lamaMFA.

> **Note**
>
> Anda tidak perlu mengganti MFA perangkat yang hilang atau dicuri dengan jenis perangkat yang sama. Misalnya, jika Anda memecahkan kunci FIDO keamanan dan memesan yang baru, Anda dapat menggunakan TOTP token virtual MFA atau perangkat keras hingga FIDO kunci baru tiba.

> **Important**
>
> Jika MFA perangkat Anda hilang atau dicuri, ubah kata sandi pengguna root Anda setelah masuk dan membuat MFA perangkat pengganti Anda. Penyerang mungkin telah mencuri perangkat otentikasi dan mungkin juga memiliki kata sandi Anda saat ini. Untuk informasi selengkapnya, lihat Ubah kata sandi untuk Pengguna root akun AWS.

Memulihkan perangkat IAM pengguna MFA

Jika Anda adalah IAM pengguna yang tidak dapat masukMFA, Anda tidak dapat memulihkan MFA perangkat sendiri. Anda harus menghubungi administrator untuk menonaktifkan perangkat. Kemudian Anda dapat mengaktifkan perangkat baru.

Untuk mendapatkan bantuan untuk MFA perangkat sebagai IAM pengguna

1. Hubungi AWS administrator atau orang lain yang memberi Anda nama pengguna dan kata sandi untuk IAM pengguna. Administrator harus menonaktifkan MFA perangkat seperti yang dijelaskan Nonaktifkan perangkat MFA sehingga Anda dapat masuk.
2. Langkah selanjutnya bervariasi tergantung pada jenis yang MFA Anda gunakan:
   - Untuk MFA perangkat virtual, hapus akun dari perangkat Anda. Kemudian aktifkan perangkat virtual seperti yang dijelaskan di Tetapkan MFA perangkat virtual di AWS Management Console.
   - Untuk kunci FIDO keamanan, hubungi penyedia pihak ketiga untuk bantuan mengganti perangkat. Saat Anda menerima kunci FIDO keamanan baru, aktifkan seperti yang dijelaskan diTetapkan kunci sandi atau kunci keamanan di AWS Management Console.
   - Untuk TOTP token perangkat keras, hubungi penyedia pihak ketiga untuk bantuan memperbaiki atau mengganti perangkat. Setelah Anda memiliki MFA perangkat fisik baru, aktifkan perangkat seperti yang dijelaskan dalamTetapkan TOTP token perangkat keras di AWS Management Console.

   > **Note**
   >
   > Anda tidak perlu mengganti MFA perangkat yang hilang atau dicuri dengan jenis perangkat yang sama. Anda dapat memiliki hingga delapan MFA perangkat kombinasi apa pun. Misalnya, jika Anda memecahkan kunci FIDO keamanan dan memesan yang baru, Anda dapat menggunakan TOTP token virtual MFA atau perangkat keras hingga FIDO kunci baru tiba.
3. Jika MFA perangkat Anda hilang atau dicuri, ubah juga kata sandi Anda jika penyerang mencuri perangkat otentikasi dan mungkin juga memiliki kata sandi Anda saat ini. Untuk informasi selengkapnya, lihat Kelola kata sandi untuk pengguna IAM

APIAkses aman dengan MFA

Dengan IAM kebijakan, Anda dapat menentukan API operasi mana yang diizinkan untuk dipanggil oleh pengguna. Anda dapat menerapkan keamanan tambahan dengan mengharuskan pengguna untuk mengautentikasi dengan otentikasi multi-faktor (MFA) sebelum Anda mengizinkan mereka melakukan tindakan yang sangat sensitif.

Misalnya, Anda mungkin memiliki kebijakan yang memungkinkan pengguna menjalankan Amazon EC2 `RunInstances``DescribeInstances`, dan `StopInstances` tindakan. Tetapi Anda mungkin ingin membatasi tindakan destruktif seperti `TerminateInstances` dan memastikan bahwa pengguna dapat melakukan tindakan itu hanya jika mereka mengautentikasi dengan perangkat. AWS MFA

Topik

Gambaran Umum

Menambahkan MFA perlindungan ke API operasi melibatkan tugas-tugas ini:

1. Administrator mengonfigurasi AWS MFA perangkat untuk setiap pengguna yang harus membuat API permintaan yang memerlukan MFA otentikasi. Untuk informasi selengkapnya, lihat AWS Otentikasi multi-faktor di IAM.
2. Administrator membuat kebijakan untuk pengguna yang menyertakan `Condition` elemen yang memeriksa apakah pengguna diautentikasi dengan AWS MFA perangkat.
3. Pengguna memanggil salah satu AWS STS API operasi yang mendukung MFA parameter: AssumeRoleatau GetSessionToken. Sebagai bagian dari panggilan, pengguna mencakup pengidentifikasi perangkat untuk perangkat yang terhubung dengan pengguna. Pengguna juga menyertakan kata sandi satu kali berbasis waktu (TOTP) yang dihasilkan perangkat. Dalam kasus lain, pengguna mendapatkan kembali kredensial keamanan sementara yang dapat digunakan pengguna untuk membuat permintaan tambahan ke AWS.

> **Note**
>
> MFAperlindungan untuk API operasi layanan hanya tersedia jika layanan mendukung kredensil keamanan sementara. Untuk daftar layanan ini, lihat Menggunakan Kredensial IT Sementara untuk Mengakses AWS.

Jika otorisasi gagal, AWS mengembalikan pesan kesalahan akses ditolak (seperti halnya untuk akses yang tidak sah). Dengan API kebijakan MFA yang dilindungi, AWS menolak akses ke API operasi yang ditentukan dalam kebijakan jika pengguna mencoba memanggil API operasi tanpa otentikasi yang validMFA. Operasi juga ditolak jika cap waktu permintaan untuk API operasi berada di luar rentang yang diizinkan yang ditentukan dalam kebijakan. Pengguna harus diautentikasi ulang MFA dengan meminta kredensil keamanan sementara baru dengan MFA kode dan nomor seri perangkat.

IAMkebijakan dengan MFA kondisi

Kebijakan dengan MFA ketentuan dapat dilampirkan sebagai berikut:

- Pengguna atau grup IAM
- Sumber daya seperti bucket Amazon S3, SQS antrian Amazon, atau topik Amazon SNS
- Kebijakan kepercayaan dari IAM peran yang dapat diasumsikan oleh pengguna

Anda dapat menggunakan MFA kondisi dalam kebijakan untuk memeriksa properti berikut:

- Keberadaan — Untuk hanya memverifikasi bahwa pengguna melakukan otentikasi denganMFA, periksa apakah `aws:MultiFactorAuthPresent` kunci dalam kondisi. `True` `Bool` Kunci tersebut hanya muncul ketika pengguna mengautentikasi dengan kredensial jangka pendek. Kredensial jangka panjang, seperti access key, tidak mencakup kunci ini.
- Durasi—Jika Anda ingin memberikan akses hanya dalam waktu tertentu setelah MFA otentikasi, gunakan tipe kondisi numerik untuk membandingkan usia `aws:MultiFactorAuthAge` kunci dengan nilai (seperti 3600 detik). Perhatikan bahwa `aws:MultiFactorAuthAge` kunci tidak MFA ada jika tidak digunakan.

Contoh berikut menunjukkan kebijakan kepercayaan IAM peran yang mencakup MFA kondisi untuk menguji keberadaan MFA otentikasi. Dengan kebijakan ini, pengguna dari `Principal` elemen yang Akun AWS ditentukan (ganti `ACCOUNT-B-ID` dengan Akun AWS ID yang valid) dapat mengambil

peran yang dilampirkan kebijakan ini. Namun pengguna tersebut hanya dapat mengambil peran jika pengguna diautentikasi menggunakanMFA.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

Untuk informasi lebih lanjut tentang jenis kondisi untukMFA, lihatAWS kunci konteks kondisi global,Operator ketentuan numerik, danOperator ketentuan memeriksa keberadaan kunci kondisi .

Memilih antara GetSessionToken dan AssumeRole

AWS STS menyediakan dua API operasi yang memungkinkan pengguna meneruskan MFA informasi: `GetSessionToken` dan`AssumeRole`. APIOperasi yang dipanggil pengguna untuk mendapatkan kredensil keamanan sementara tergantung pada skenario berikut yang berlaku.

Gunakan **`GetSessionToken`** untuk skenario berikut:

- APIOperasi panggilan yang mengakses sumber daya Akun AWS sama dengan IAM pengguna yang membuat permintaan. Perhatikan bahwa kredensil sementara dari `GetSessionToken` permintaan dapat mengakses IAM dan AWS STS API beroperasi hanya jika Anda menyertakan MFA informasi dalam permintaan kredensil. Karena kredensil sementara yang dikembalikan oleh `GetSessionToken` menyertakan MFA informasi, Anda dapat memeriksa MFA dalam API operasi individual yang dibuat oleh kredensialnya.
- Akses ke sumber daya yang dilindungi dengan kebijakan berbasis sumber daya yang mencakup suatu kondisi. MFA

Tujuan dari `GetSessionToken` operasi ini adalah untuk mengotentikasi pengguna menggunakanMFA. Anda tidak dapat menggunakan kebijakan untuk mengontrol operasi autentikasi.

Gunakan **`AssumeRole`** untuk skenario berikut:

- APIOperasi panggilan yang mengakses sumber daya dalam hal yang sama atau berbeda Akun AWS. APIPanggilan dapat mencakup salah satu IAM atau AWS STS API. Perhatikan bahwa

untuk melindungi akses, Anda menerapkan MFA pada saat pengguna mengambil peran tersebut. Kredensi sementara yang dikembalikan oleh `AssumeRole` tidak menyertakan MFA informasi dalam konteks, sehingga Anda tidak dapat memeriksa API operasi individu untuk. MFA Inilah mengapa Anda harus menggunakan `GetSessionToken` untuk membatasi akses ke sumber daya yang dilindungi oleh kebijakan berbasis sumber daya.

Perincian tentang cara menerapkan skenario ini akan dijelaskan nanti dalam dokumen ini.

Poin penting tentang MFA akses yang dilindungi API

Penting untuk memahami aspek-aspek MFA perlindungan berikut untuk API operasi:

- MFAperlindungan hanya tersedia dengan kredensil keamanan sementara, yang harus diperoleh dengan `AssumeRole` atau. `GetSessionToken`
- Anda tidak dapat menggunakan API akses MFA yang dilindungi dengan Pengguna root akun AWS kredensil.
- Anda tidak dapat menggunakan API akses MFA yang dilindungi dengan kunci keamanan U2F.
- Pengguna federasi tidak dapat diberikan MFA perangkat untuk digunakan dengan AWS layanan, sehingga mereka tidak dapat mengakses AWS sumber daya yang dikendalikan olehMFA. (Lihat poin berikutnya.)
- AWS STS APIOperasi lain yang mengembalikan kredensi sementara tidak mendukung. MFA Untuk `AssumeRoleWithWebIdentity` dan`AssumeRoleWithSAML`, pengguna diautentikasi oleh penyedia eksternal dan AWS tidak dapat menentukan apakah penyedia itu diperlukanMFA. Sebab`GetFederationToken`, MFA belum tentu terkait dengan pengguna tertentu.
- Demikian pula, kredensi jangka panjang (kunci akses IAM pengguna dan kunci akses pengguna root) tidak dapat digunakan dengan API akses MFA yang dilindungi karena tidak kedaluwarsa.
- `AssumeRole`dan juga `GetSessionToken` dapat dipanggil tanpa MFA informasi. Dalam hal ini, penelepon mendapatkan kembali kredensil keamanan sementara, tetapi informasi sesi untuk kredensil sementara tersebut tidak menunjukkan bahwa pengguna diautentikasi dengan. MFA
- Untuk menetapkan MFA perlindungan untuk API operasi, Anda menambahkan MFA kondisi ke kebijakan. Kebijakan harus menyertakan kunci `aws:MultiFactorAuthPresent` kondisi untuk menegakkan penggunaan. MFA Untuk pendelegasian lintas akun, kebijakan kepercayaan peran tersebut harus mencakup kunci ketentuan.
- Ketika Anda Akun AWS mengizinkan orang lain mengakses sumber daya di akun Anda, keamanan sumber daya Anda tergantung pada konfigurasi akun tepercaya (akun lain, bukan milik Anda). Hal ini tetap berlaku meskipun Anda memerlukan autentikasi multi-faktor. Identitas apa pun di akun

tepercaya yang memiliki izin untuk membuat MFA perangkat virtual dapat membuat MFA klaim untuk memenuhi bagian dari kebijakan kepercayaan peran Anda. Sebelum Anda mengizinkan anggota akun lain mengakses AWS sumber daya Anda yang memerlukan otentikasi multi-faktor, Anda harus memastikan bahwa pemilik akun tepercaya mengikuti praktik terbaik keamanan. Misalnya, akun tepercaya harus membatasi akses ke API operasi sensitif, seperti API operasi MFA manajemen perangkat, ke identitas tertentu dan tepercaya.

- Jika kebijakan menyertakan MFA kondisi, permintaan ditolak jika pengguna belum MFA diautentikasi, atau jika mereka memberikan pengenal MFA perangkat yang tidak valid atau tidak valid. TOTP

Skenario: MFA perlindungan untuk delegasi lintas akun

Dalam skenario ini, Anda ingin mendelegasikan akses ke IAM pengguna di akun lain, tetapi hanya jika pengguna diautentikasi dengan perangkat. AWS MFA (Untuk informasi lebih lanjut tentang pendelegasian lintas akun, lihat Istilah dan konsep peran.

Bayangkan Anda memiliki akun A (akun kepercayaan yang memiliki sumber daya untuk diakses), dengan IAM pengguna Anaya, yang memiliki izin administrator. Dia ingin memberikan akses ke pengguna Richard di akun B (akun tepercaya), tetapi ingin memastikan bahwa Richard diautentikasi MFA sebelum dia mengambil peran.

1. Dalam akun kepercayaan A, Anaya membuat IAM peran bernama `CrossAccountRole` dan menetapkan prinsipal dalam kebijakan kepercayaan peran ke ID akun B. Kebijakan kepercayaan memberikan izin untuk tindakan tersebut. AWS STS `AssumeRole` Anaya juga menambahkan MFA syarat pada kebijakan kepercayaan, seperti pada contoh berikut.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

2. Anaya menambahkan kebijakan izin untuk peran yang menentukan apa yang dapat dilakukan peran tersebut. Kebijakan izin untuk peran dengan MFA perlindungan tidak berbeda dengan kebijakan izin peran lainnya. Contoh berikut menunjukkan kebijakan yang ditambahkan

Anaya ke peran; ini memungkinkan pengguna yang berasumsi untuk melakukan tindakan Amazon DynamoDB apa pun pada `Books` tabel di akun A. Kebijakan ini juga mengizinkan `dynamodb:ListTables` tindakan, yang diperlukan untuk melakukan tindakan di konsol.

> **Note**
>
> Kebijakan izin tidak menyertakan MFA kondisi. Penting untuk dipahami bahwa MFA otentikasi hanya digunakan untuk menentukan apakah pengguna dapat mengambil peran tersebut. Setelah pengguna mengambil peran, tidak ada MFA pemeriksaan lebih lanjut yang dilakukan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:ACCOUNT-A-ID:table/Books"
        },
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
}
```

3. Di akun B tepercaya, administrator memastikan bahwa IAM pengguna Richard dikonfigurasi dengan AWS MFA perangkat dan bahwa ia mengetahui ID perangkat. ID perangkat adalah nomor seri jika itu adalah MFA perangkat keras, atau perangkat ARN jika itu MFA perangkat virtual.
4. Di akun B, administrator melampirkan kebijakan berikut kepada pengguna Richard (atau grup yang ia anggota) yang memungkinkannya untuk menghubungi tindakan `AssumeRole`. Sumber daya diatur ke peran ARN yang dibuat Anaya pada langkah 1. Perhatikan bahwa kebijakan ini tidak mengandung MFA kondisi.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [{
    "Effect": "Allow",
    "Action": ["sts:AssumeRole"],
    "Resource": ["arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole"]
  }]
}
```

5. Di akun B, Richard (atau aplikasi yang dijalankan Richard) akan memanggil `AssumeRole`. APIPanggilan tersebut ARN mencakup peran untuk asumsi (`arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole`), ID MFA perangkat, dan arus TOTP yang diperoleh Richard dari perangkatnya.

   Ketika Richard meneleponAssumeRole, AWS menentukan apakah dia memiliki kredensi yang valid, termasuk persyaratan untuk. MFA Jika demikian, Richard berhasil mengasumsikan peran tersebut dan dapat melakukan tindakan DynamoDB apa pun pada tabel yang `Books` disebutkan di akun A saat menggunakan kredenal sementara peran.

   Sebagai contoh program yang memanggil `AssumeRole`, lihat Memanggil AssumeRole dengan MFA otentikasi.

Skenario: MFA perlindungan untuk akses ke API operasi di akun saat ini

Dalam skenario ini, Anda harus memastikan bahwa pengguna di Anda Akun AWS dapat mengakses API operasi sensitif hanya ketika pengguna diautentikasi menggunakan AWS MFA perangkat.

Bayangkan Anda memiliki akun A yang berisi sekelompok pengembang yang perlu bekerja dengan EC2 instance. Developer biasa dapat bekerja dengan instans, tetapi mereka tidak diberikan izin tindakan `ec2:StopInstances` atau `ec2:TerminateInstances`. Anda ingin membatasi tindakan istimewa “destruktif” tersebut hanya untuk beberapa pengguna tepercaya, sehingga Anda menambahkan MFA perlindungan pada kebijakan yang memungkinkan tindakan Amazon EC2 yang sensitif ini.

Dalam skenario ini, salah satu pengguna tepercaya tersebut adalah pengguna Sofía. Pengguna Anaya adalah administrator di akun A.

1. Anaya memastikan bahwa Sofía dikonfigurasi dengan AWS MFA perangkat dan Sofía mengetahui ID perangkat tersebut. ID perangkat adalah nomor seri jika itu adalah MFA perangkat keras, atau perangkat ARN jika itu MFA perangkat virtual.
2. Anaya membuat kelompok bernama `EC2-Admins` dan menambahkan Sofía ke grup.

3. Anaya melampirkan kebijakan berikut ke kelompok `EC2-Admins`. Kebijakan ini memberi pengguna izin untuk memanggil Amazon EC2 `StopInstances` dan `TerminateInstances` tindakan hanya jika pengguna telah mengautentikasi penggunaan. MFA

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:StopInstances",
      "ec2:TerminateInstances"
    ],
    "Resource": ["*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

4.

> **Note**
>
> Agar kebijakan ini berlaku, pengguna harus keluar terlebih dahulu dan masuk kembali.

   Jika pengguna Sofía perlu menghentikan atau menghentikan EC2 instance Amazon, dia (atau aplikasi yang sedang dia jalankan) memanggil. `GetSessionToken` APIOperasi ini melewati ID MFA perangkat dan arus TOTP yang didapat Sofía dari perangkatnya.

5. Pengguna Sofía (atau aplikasi yang digunakan Sofía) menggunakan kredensil sementara yang disediakan oleh untuk `GetSessionToken` memanggil Amazon atau tindakan. EC2 `StopInstances` `TerminateInstances`

   Sebagai contoh program yang memanggil `GetSessionToken`, lihat Memanggil GetSessionToken dengan MFA otentikasi nanti di dokumen ini.

Skenario: MFA perlindungan untuk sumber daya yang memiliki kebijakan berbasis sumber daya

Dalam skenario ini, Anda adalah pemilik bucket S3, SQS antrian, atau topik. SNS Anda ingin memastikan bahwa setiap pengguna dari Akun AWS siapa pun yang mengakses sumber daya diautentikasi oleh perangkat. AWS MFA

Skenario ini menggambarkan cara untuk memberikan MFA perlindungan lintas akun tanpa mengharuskan pengguna untuk mengambil peran terlebih dahulu. Dalam hal ini, pengguna dapat mengakses sumber daya jika tiga kondisi terpenuhi: Pengguna harus diautentikasi olehMFA, dapat memperoleh kredensil keamanan sementara dari`GetSessionToken`, dan berada di akun yang dipercaya oleh kebijakan sumber daya.

Bayangkan Anda berada di akun A dan Anda membuat bucket S3. Anda ingin memberikan akses ke bucket ini kepada pengguna yang berbeda Akun AWS, tetapi hanya jika pengguna tersebut diautentikasiMFA.

Dalam skenario ini, pengguna Anaya adalah administrator di akun A. Pengguna Nikhil adalah IAM pengguna di akun C.

1. Pada akun A, Anaya membuat bucket dengan nama `Account-A-bucket`.
2. Anaya menambahkan kebijakan bucket ke bucket. Kebijakan ini memungkinkan pengguna di akun A, akun B, atau akun C untuk melakukan tindakan Amazon S3 `PutObject` dan `DeleteObject` di dalam bucket. Kebijakan tersebut mencakup suatu MFA kondisi.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> Amazon S3 menawarkan fitur MFA Hapus untuk akses akun root (hanya). Anda dapat mengaktifkan Amazon S3 MFA Delete saat menyetel status pembuatan versi bucket.

Amazon S3 MFA Delete tidak dapat diterapkan ke IAM pengguna, dan dikelola secara independen dari akses MFA yang API dilindungi. IAMPengguna dengan izin untuk menghapus bucket tidak dapat menghapus bucket dengan Amazon MFA S3 Delete diaktifkan. Untuk informasi selengkapnya tentang Hapus Amazon S3, lihat MFA MFA Menghapus.

3. Di akun C, administrator memastikan bahwa pengguna Nikhil dikonfigurasi dengan AWS MFA perangkat dan bahwa ia mengetahui ID perangkat. ID perangkat adalah nomor seri jika itu adalah MFA perangkat keras, atau perangkat ARN jika itu MFA perangkat virtual.
4. Dalam akun C, Nikhil (atau aplikasi yang dia jalankan) akan memanggil `GetSessionToken`. Panggilan tersebut mencakup ID atau ARN MFA perangkat dan arus TOTP yang diperoleh Nikhil dari perangkatnya.
5. Nikhil (atau aplikasi yang dia gunakan) menggunakan kredensial sementara yang dikembalikan oleh `GetSessionToken` untuk menghubungi tindakan Amazon S3 `PutObject` untuk mengunggah file ke `Account-A-bucket`.

   Sebagai contoh program yang memanggil `GetSessionToken`, lihat Memanggil GetSessionToken dengan MFA otentikasi nanti di dokumen ini.

> **Note**
>
> Kredensial sementara yang dikembalikan `AssumeRole` tidak akan bekerja dalam kasus ini. Meskipun pengguna dapat memberikan MFA informasi untuk mengambil peran, kredenal sementara yang dikembalikan oleh `AssumeRole` tidak menyertakan informasi tersebutMFA. Informasi itu diperlukan untuk memenuhi MFA kondisi dalam kebijakan.

Kode sampel: Meminta kredensial dengan autentikasi multi-faktor

Contoh berikut menunjukkan cara memanggil `GetSessionToken` dan `AssumeRole` operasi dan meneruskan parameter MFA otentikasi. Tidak ada izin yang diperlukan untuk memanggil `GetSessionToken`, tetapi Anda harus memiliki kebijakan yang mengijinkan Anda untuk memanggil `AssumeRole`. Kredensial yang dihasilkan kemudian digunakan untuk mencantumkan semua bucket S3 di akun.

Memanggil GetSessionToken dengan MFA otentikasi

Contoh berikut menunjukkan cara memanggil `GetSessionToken` dan meneruskan informasi MFA otentikasi. Kredensial keamanan sementara yang dihasilkan oleh operasi `GetSessionToken` kemudian digunakan untuk mencantumkan semua bucket S3 di akun.

Kebijakan yang terlampir pada pengguna yang menjalankan kode ini (atau ke grup yang ada di pengguna) memberikan izin untuk kredensial sementara yang dikembalikan. Untuk contoh kode ini, kebijakan harus memberikan izin kepada pengguna untuk meminta operasi Amazon S3 `ListBuckets`.

Contoh kode berikut menunjukkan cara menggunakan`GetSessionToken`.

CLI

AWS CLI

Untuk mendapatkan satu set kredensi jangka pendek untuk identitas IAM

`get-session-token`Perintah berikut mengambil satu set kredensi jangka pendek untuk IAM identitas yang membuat panggilan. Kredensi yang dihasilkan dapat digunakan untuk permintaan di mana otentikasi multi-faktor (MFA) diperlukan oleh kebijakan. Kredensialnya kedaluwarsa 15 menit setelah dibuat.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Output:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat Meminta Kredensial Keamanan Sementara di Panduan Pengguna.AWS IAM

- Untuk API detailnya, lihat GetSessionTokendi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan sebuah **Amazon.RuntimeAWSCredentials** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensil sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensil lainnya, gunakan parameter - ProfileName atau - AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 2: Mengembalikan sebuah **Amazon.RuntimeAWSCredentials** instance yang berisi kredensial sementara yang valid selama satu jam. Kredensi yang digunakan untuk membuat permintaan diperoleh dari profil yang ditentukan.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 3: Mengembalikan **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredenal sementara yang valid selama satu jam menggunakan nomor identifikasi MFA perangkat yang terkait dengan akun yang kredensialnya ditentukan dalam profil 'myprofilename' dan nilai yang disediakan oleh perangkat.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Untuk API detailnya, lihat GetSessionTokendi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan token sesi dengan meneruskan MFA token dan gunakan untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.
```

```
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk API detailnya, lihat GetSessionToken AWSSDKReferensi Python (Boto3). API

Memanggil AssumeRole dengan MFA otentikasi

Contoh berikut menunjukkan cara memanggil `AssumeRole` dan meneruskan informasi MFA otentikasi. Kredensial keamanan sementara yang dikembalikan oleh `AssumeRole` kemudian digunakan untuk mencantumkan semua bucket Amazon S3 di akun.

Untuk informasi selengkapnya tentang skenario ini, lihat Skenario: MFA perlindungan untuk delegasi lintas akun.

Contoh kode berikut menunjukkan cara menggunakan`AssumeRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
```

```
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
```

```
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        ech o"Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat AssumeRoledi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mengambil peran

`assume-role`Perintah berikut mengambil satu set kredensi jangka pendek untuk peran tersebut. IAM `s3-access-example`

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Output:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
```

```
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

Output dari perintah berisi kunci akses, kunci rahasia, dan token sesi yang dapat Anda gunakan untuk AWS mengautentikasi.

Untuk AWS CLI digunakan, Anda dapat mengatur profil bernama yang terkait dengan peran. Ketika Anda menggunakan profil, AWS CLI akan memanggil assume-role dan mengelola kredensi untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan IAM peran AWS CLI dalam Panduan AWS CLI Pengguna.

- Untuk API detailnya, lihat AssumeRoledi Referensi AWS CLI Perintah.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
```

```
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
```

```
                    roleSessionName - An identifier for the assumed role session
(for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
exTime);
```

```
        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat klien.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Asumsikan IAM perannya.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
```

```
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });
```

```
//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan satu set kredensi sementara (kunci akses, kunci rahasia, dan token sesi) yang dapat digunakan selama satu jam untuk mengakses AWS sumber daya yang biasanya tidak dapat diakses oleh pengguna yang meminta. Kredensi yang dikembalikan memiliki izin yang diizinkan oleh kebijakan akses dari peran yang diasumsikan dan kebijakan yang diberikan (Anda tidak dapat menggunakan kebijakan yang disediakan untuk memberikan izin melebihi yang ditentukan oleh kebijakan akses peran yang diasumsikan).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Contoh 2: Mengembalikan satu set kredensi sementara, berlaku selama satu jam, yang memiliki izin yang sama yang ditentukan dalam kebijakan akses peran yang diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Contoh 3: Mengembalikan satu set kredensi sementara yang memasok nomor seri dan token yang dihasilkan dari kredensial yang MFA terkait dengan pengguna yang digunakan untuk mengeksekusi cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Contoh 4: Mengembalikan satu set kredensi sementara yang telah mengambil peran yang ditentukan dalam akun pelanggan. Untuk setiap peran yang dapat diasumsikan oleh pihak ketiga, akun pelanggan harus membuat peran menggunakan pengidentifikasi yang harus diteruskan dalam ExternalId parameter - setiap kali peran diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Untuk API detailnya, lihat AssumeRoledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Asumsikan IAM peran yang memerlukan MFA token dan gunakan kredensil sementara untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk API detailnya, lihat AssumeRole AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Untuk API detailnya, lihat AssumeRole AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat AssumeRole AWSSDKAPIreferensi Swift.

Temukan kredensi yang tidak digunakan AWS

Untuk meningkatkan keamanan Anda Akun AWS, hapus kredensi pengguna IAM (yaitu, kata sandi dan kunci akses) yang tidak diperlukan. Misalnya, ketika pengguna meninggalkan organisasi Anda atau tidak lagi memerlukan AWS akses, temukan kredensi yang mereka gunakan dan pastikan bahwa mereka tidak lagi beroperasi. Idealnya, Anda menghapus kredensial jika tidak lagi diperlukan. Anda dapat selalu membuat ulang catatan di kemudian hari jika perlu. Sekurang-kurangnya, Anda harus mengubah kata sandi atau menonaktifkan access key sehingga pengguna sebelumnya tidak lagi memiliki akses.

Tentu saja, definisi dari tidak digunakan dapat bervariasi dan biasanya berarti kredensial yang belum digunakan dalam jangka waktu tertentu.

Menemukan kata sandi yang tidak digunakan

Anda dapat menggunakan AWS Management Console untuk melihat informasi penggunaan kata sandi untuk pengguna Anda. Jika Anda memiliki sejumlah besar pengguna, Anda dapat menggunakan konsol untuk mengunduh laporan kredensial dengan informasi tentang kapan terakhir pengguna menggunakan kata sandi konsol mereka. Anda juga dapat mengakses informasi dari AWS CLI atau API IAM.

Untuk menemukan yang tidak digunakan (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan Masuk terakhir konsol pada tabel pengguna:
   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ().
   b. Di Pilih kolom yang terlihat, pilih Konsol masuk terakhir.
   c. Pilih Konfirmasi untuk kembali ke daftar pengguna.
4. Kolom login terakhir Konsol menunjukkan tanggal saat pengguna terakhir kali masuk AWS melalui konsol. Anda dapat menggunakan informasi ini untuk menemukan kata sandi pengguna yang tidak masuk lebih dari periode waktu tertentu. Kolom menampilkan Tidak pernah bagi pengguna dengan kata sandi yang belum pernah masuk. Tidak ada menunjukkan pengguna

tanpa kata sandi. Kata sandi yang belum digunakan baru-baru ini mungkin merupakan calon yang baik untuk ditiadakan.

> **Important**
>
> Karena masalah layanan, data penggunaan terakhir kata sandi tidak mencakup penggunaan kata sandi dari tanggal 3 Mei 2018 22.50 PDT hingga 23 Mei 2018 14.08 PDT. Hal ini memengaruhi tanggal login terakhir yang ditampilkan di konsol IAM dan kata sandi tanggal terakhir digunakan dalam laporan kredensi IAM, dan dikembalikan oleh operasi API. GetUser Jika pengguna masuk selama waktu yang terpengaruh, tanggal terakhir kali menggunakan kata sandi yang dikembalikan adalah tanggal pengguna terakhir masuk sebelum 3 Mei 2018. Untuk pengguna yang masuk setelah 23 Mei 2018 14.08 PDT, tanggal penggunaan terakhir kata sandi yang dikembalikan adalah akurat. Jika Anda menggunakan kata sandi informasi yang terakhir digunakan untuk mengidentifikasi kredensi yang tidak digunakan untuk penghapusan, seperti menghapus pengguna yang tidak masuk dalam 90 hari terakhir, kami sarankan Anda menyesuaikan jendela evaluasi Anda untuk menyertakan tanggal setelah 23 Mei 2018. AWS Atau, jika pengguna Anda menggunakan kunci akses untuk mengakses AWS secara terprogram, Anda dapat merujuk ke kunci akses informasi yang terakhir digunakan karena akurat untuk semua tanggal.

Untuk menemukan kata sandi yang tidak digunakan dengan mengunduh laporan kredensial (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Laporan kredensial.
3. Pilih Unduh Laporan untuk mengunduh file CSV (nilai yang dipisahkan koma) bernama `status_reports_<date>T<time>.csv`. Kolom kelima berisi `password_last_used` kolom dengan tanggal atau salah satu dari berikut:
   - N/A – Pengguna yang tidak memiliki kata sandi yang ditetapkan sama sekali.
   - no_information – Pengguna yang belum menggunakan kata sandi mereka sejak IAM mulai melacak umur kata sandi pada 20 Oktober 2014.

Untuk menemukan kata sandi yang tidak digunakan (AWS CLI)

Jalankan perintah berikut untuk menemukan kata sandi yang tidak digunakan:

- `aws iam list-users` mengembalikan daftar pengguna, masing-masing dengan nilai `PasswordLastUsed`. Jika nilai hilang, maka pengguna tidak memiliki kata sandi atau kata sandi belum digunakan karena IAM mulai melacak usia kata sandi pada 20 Oktober 2014.

Untuk menemukan kata sandi yang tidak digunakan (AWS API)

Hubungi operasi berikut untuk menemukan kata sandi yang tidak digunakan:

- `ListUsers` mengembalikan koleksi pengguna, masing-masing memiliki nilai `<PasswordLastUsed>`. Jika nilai hilang, maka pengguna tidak memiliki kata sandi atau kata sandi belum digunakan karena IAM mulai melacak usia kata sandi pada 20 Oktober 2014.

Untuk informasi tentang perintah untuk mengunduh laporan kredensial, lihat Mendapatkan laporan kredensial (AWS CLI).

Menemukan access key yang tidak digunakan

Anda dapat menggunakan AWS Management Console untuk melihat informasi penggunaan kunci akses bagi pengguna Anda. Jika Anda memiliki sejumlah besar pengguna, Anda dapat menggunakan konsol untuk mengunduh laporan kredensial tentang kapan terakhir pengguna menggunakan access key mereka. Anda juga dapat mengakses informasi dari AWS CLI atau API IAM.

Untuk menemukan access key yang tidak digunakan (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan kolom Access key terakhir digunakan pada tabel pengguna:
   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ( ).
   b. Di Pilih kolom yang terlihat, pilih Kunci akses terakhir digunakan.
   c. Pilih Konfirmasi untuk kembali ke daftar pengguna.

4. Kolom kunci Access terakhir digunakan menunjukkan jumlah hari sejak pengguna terakhir diakses secara AWS terprogram. Anda dapat menggunakan informasi ini untuk menemukan kata sandi pengguna yang tidak digunakan lebih dari periode waktu tertentu. Kolom menampilkan - untuk pengguna tanpa kunci akses. Access key yang belum digunakan baru-baru ini mungkin merupakan calon yang baik untuk ditiadakan.

Untuk menemukan access key yang tidak digunakan dengan mengunduh laporan kredensial (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Laporan Kredensial.
3. Pilih Unduh Laporan untuk mengunduh file CSV (nilai yang dipisahkan koma) bernama `status_reports_<date>T<time>.csv`. Kolom 11 sampai 13 berisi tanggal terakhir digunakan, Wilayah, dan layanan informasi untuk access key 1. Kolom 16 sampai 18 berisi informasi yang sama untuk access key 2. Nilainya adalah N/A jika pengguna tidak memiliki kunci akses atau pengguna belum menggunakan kunci akses sejak IAM mulai melacak usia kunci akses pada 22 April 2015.

Untuk menemukan access key yang tidak digunakan (AWS CLI)

Jalankan perintah berikut untuk menemukan access key yang tidak digunakan:

- `aws iam list-access-keys` mengembalikan informasi tentang access key untuk pengguna, termasuk `AccessKeyID`.
- `aws iam get-access-key-last-used` mengambil access key ID dan hasil pengembalian yang mencakup `LastUsedDate`, `Region` yang access key-nya terakhir digunakan, dan `ServiceName` dari layanan terakhir yang diminta. Jika `LastUsedDate` hilang, maka kunci akses belum digunakan sejak IAM mulai melacak usia kunci akses pada 22 April 2015.

Untuk menemukan kunci akses yang tidak digunakan (AWS API)

Hubungi operasi berikut untuk menemukan access key yang tidak digunakan:

- `ListAccessKeys` mengembalikan daftar dari nilai `AccessKeyID` untuk access key yang terkait dengan pengguna yang ditentukan.
- `GetAccessKeyLastUsed` mengambil access key ID dan mengembalikan kumpulan nilai. Yang disertakan adalah `LastUsedDate`, `Region` di mana access key terakhir digunakan, dan

`ServiceName` dari layanan terakhir yang diminta. Jika nilai hilang, maka pengguna tidak memiliki kunci akses atau kunci akses tidak digunakan sejak IAM mulai melacak usia kunci akses pada 22 April 2015.

Untuk informasi tentang perintah untuk mengunduh laporan kredensial, lihat Mendapatkan laporan kredensial (AWS CLI).

Hasilkan laporan kredensi untuk Anda Akun AWS

Anda dapat membuat dan mengunduh laporan kredensi yang mencantumkan semua pengguna di akun Anda dan status berbagai kredensialnya, termasuk kata sandi, kunci akses, dan perangkat. MFA Anda bisa mendapatkan laporan kredensi dari AWS Management Console, Alat Baris Perintah AWS SDKsdan, atau. IAM API

Anda dapat menggunakan laporan kredensial untuk membantu upaya audit dan kepatuhan Anda. Anda dapat menggunakan laporan untuk mengaudit efek persyaratan siklus hidup kredenal, seperti kata sandi dan pembaruan kunci akses. Anda dapat memberikan laporan ke audit eksternal, atau memberikan izin kepada auditor sehingga dia dapat mengunduh laporan secara langsung.

Anda dapat membuat laporan kredensial sesering sekali setiap empat jam. Saat Anda meminta laporan, periksa IAM terlebih dahulu apakah laporan untuk laporan Akun AWS telah dibuat dalam empat jam terakhir. Jika demikian, laporan terbaru diunduh. Jika laporan terbaru untuk akun tersebut lebih dari empat jam, atau jika tidak ada laporan sebelumnya untuk akun tersebut, IAM buat dan unduh laporan baru.

Topik

- Izin yang diperlukan
- Memahami format laporan
- Mendapatkan laporan kredensial (konsol)
- Mendapatkan laporan kredensial (AWS CLI)
- Mendapatkan laporan kredensi ()AWS API

Izin yang diperlukan

Izin berikut diperlukan untuk membuat dan mengunduh laporan:

- Untuk membuat laporan kredensial: `iam:GenerateCredentialReport`

- Untuk mengunduh laporan: `iam:GetCredentialReport`

Memahami format laporan

Laporan kredensi diformat sebagai file value () yang dipisahkan koma. CSV Anda dapat membuka CSV file dengan perangkat lunak spreadsheet umum untuk melakukan analisis, atau Anda dapat membangun aplikasi yang mengkonsumsi CSV file secara terprogram dan melakukan analisis khusus.

CSVFile berisi kolom berikut:

user

Nama pengguna yang ramah.

arn

Nama Sumber Daya Amazon (ARN) pengguna. Untuk informasi lebih lanjut tentangARNs, lihatIAM ARNs.

user_creation_time

Tanggal dan waktu ketika pengguna dibuat, dalam format tanggal-waktu ISO 8601.

password_enabled

Ketika pengguna memiliki kata sandi, nilai ini adalah `TRUE`. Jika tidak, nilainya `FALSE`.

password_last_used

Tanggal dan waktu ketika kata sandi Pengguna root akun AWS atau pengguna terakhir digunakan untuk masuk ke AWS situs web, dalam format tanggal-waktu ISO 8601. AWS situs web yang menangkap waktu masuk terakhir pengguna adalah AWS Management Console, Forum AWS Diskusi, dan AWS Marketplace. Ketika kata sandi digunakan lebih dari sekali dalam rentang 5 menit, hanya penggunaan pertama yang dicatat dalam bidang ini.

- Nilai dalam kolom ini adalah no`_information` dalam kasus ini:
  - Kata sandi pengguna belum pernah digunakan.
  - Tidak ada data login yang terkait dengan kata sandi, seperti ketika kata sandi pengguna belum digunakan setelah IAM mulai melacak informasi ini pada 20 Oktober 2014.
- Nilai dalam kolom ini adalah `N/A` (tidak berlaku) jika pengguna tidak memiliki kata sandi.

> **Important**
>
> Karena masalah layanan, kata sandi data yang terakhir digunakan tidak termasuk penggunaan kata sandi dari 3 Mei 2018 22:50 PDT hingga 23 Mei 2018 14:08. PDT Ini memengaruhi tanggal masuk terakhir yang ditampilkan di IAM konsol dan kata sandi tanggal terakhir digunakan dalam laporan IAM kredensi, dan dikembalikan oleh operasi. GetUser API Jika pengguna masuk selama waktu yang terpengaruh, tanggal terakhir kali menggunakan kata sandi yang dikembalikan adalah tanggal pengguna terakhir masuk sebelum 3 Mei 2018. Untuk pengguna yang masuk setelah 23 Mei 2018 14:08PDT, kata sandi yang dikembalikan tanggal terakhir digunakan adalah akurat.
> Jika Anda menggunakan kata sandi informasi yang terakhir digunakan untuk mengidentifikasi kredensi yang tidak digunakan untuk penghapusan, seperti menghapus pengguna yang tidak masuk dalam 90 hari terakhir, kami sarankan Anda menyesuaikan jendela evaluasi Anda untuk menyertakan tanggal setelah 23 Mei 2018. AWS Atau, jika pengguna Anda menggunakan kunci akses untuk mengakses AWS secara terprogram, Anda dapat merujuk ke kunci akses informasi yang terakhir digunakan karena akurat untuk semua tanggal.

password_last_changed

Tanggal dan waktu ketika kata sandi pengguna terakhir ditetapkan, dalam format tanggal-waktu ISO 8601. Jika pengguna tidak memiliki kata sandi, nilai dalam kolom ini adalah `N/A` (tidak berlaku).

password_next_rotation

Ketika akun memiliki kebijakan kata sandi yang memerlukan rotasi kata sandi, bidang ini berisi tanggal dan waktu, dalam format tanggal-waktu ISO 8601, ketika pengguna diminta untuk mengatur kata sandi baru. Nilai untuk Akun AWS (root) selalu`not_supported`.

mfa_active

Ketika perangkat otentikasi multi-faktor (MFA) telah diaktifkan untuk pengguna, nilai ini adalah. `TRUE` Jika tidak, nilainya `FALSE`.

access_key_1_active

Saat pengguna memiliki access key dan status access key `Active`, nilainya `TRUE` Jika tidak, nilainya `FALSE`. Berlaku untuk pengguna root akun dan IAM pengguna.

access_key_1_last_rotated

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, ketika kunci akses pengguna dibuat atau terakhir diubah. Jika pengguna tidak memiliki access key aktif, nilai dalam kolom ini adalah `N/A` (tidak berlaku). Berlaku untuk pengguna root akun dan IAM pengguna.

akses_key_1_last_used_date

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, ketika kunci akses pengguna paling baru digunakan untuk menandatangani permintaan. AWS API Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan IAM pengguna.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key.
- Access key belum pernah digunakan.
- Kunci akses belum digunakan setelah IAM mulai melacak informasi ini pada 22 April 2015.

akses_key_1_last_used_region

Wilayah AWS yang access key-nya baru-baru ini digunakan. Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan IAM pengguna.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key.
- Access key belum pernah digunakan.
- Kunci akses terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.
- Layanan yang terakhir digunakan tidak spesifik Wilayah, seperti Amazon S3

akses_key_1_last_used_service

AWS Layanan yang paling baru diakses dengan kunci akses. Nilai di bidang ini menggunakan ruang nama layanan—misalnya, untuk Amazon `s3` S3 dan Amazon. `ec2` EC2 Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan IAM pengguna.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key.

- Access key belum pernah digunakan.
- Kunci akses terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.

access_key_2_active

Saat pengguna memiliki access key kedua dan status access key kedua adalah `Active`, nilainya `TRUE`. Jika tidak, nilainya `FALSE`. Berlaku untuk pengguna root akun dan IAM pengguna.

> **Note**
>
> Pengguna dapat memiliki hingga dua tombol akses, untuk mempermudah rotasi dengan memperbarui kunci terlebih dahulu dan kemudian menghapus kunci sebelumnya. Untuk informasi selengkapnya tentang memperbarui kunci akses, lihatPerbarui kunci akses.

access_key_2_last_rotated

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, ketika kunci akses kedua pengguna dibuat atau terakhir diperbarui. Jika pengguna tidak memiliki access key aktif kedua, nilai dalam bidang ini adalah `N/A` (tidak berlaku). Berlaku untuk pengguna root akun dan IAM pengguna.

akses_key_2_last_used_date

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, ketika kunci akses kedua pengguna paling baru digunakan untuk menandatangani permintaan. AWS API Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan IAM pengguna.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key kedua.
- Access key kedua pengguna belum pernah digunakan.
- Kunci akses kedua pengguna terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.

akses_key_2_last_used_region

Wilayah AWS yang access key keduanya baru-baru ini digunakan. Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan IAM pengguna. Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key kedua.
- Access key kedua pengguna belum pernah digunakan.
- Kunci akses kedua pengguna terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.
- Layanan yang terakhir digunakan tidak spesifik Wilayah, seperti Amazon S3

access_key_2_last_used_service

AWS Layanan yang paling baru diakses dengan kunci akses kedua pengguna. Nilai di bidang ini menggunakan ruang nama layanan—misalnya, untuk Amazon `s3` S3 dan Amazon. `ec2` EC2 Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan IAM pengguna. Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key kedua.
- Access key kedua pengguna belum pernah digunakan.
- Kunci akses kedua pengguna terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.

cert_1_active

Saat pengguna memiliki sertifikat tanda tangan X.509 dan status sertifikat tersebut adalah `Active`, nilainya adalah `TRUE`. Jika tidak, nilainya `FALSE`.

cert_1_last_rotated

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, saat sertifikat penandatanganan pengguna dibuat atau terakhir diubah. Apabila pengguna tidak memiliki sertifikat penandatanganan aktif, nilai dalam kolom ini adalah `N/A` (tidak berlaku).

cert_2_active

Saat pengguna memiliki sertifikat tanda tangan X.509 kedua dan status sertifikat tersebut adalah `Active`, nilainya adalah `TRUE`. Jika tidak, nilainya `FALSE`.

> **Note**
>
> Pengguna dapat memiliki hingga dua sertifikat tanda tangan X.509, untuk membuat rotasi sertifikat lebih mudah.

cert_2_last_rotated

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, ketika sertifikat penandatanganan kedua pengguna dibuat atau terakhir diubah. Apabila pengguna tidak memiliki sertifikat tanda tangan aktif kedua, nilai dalam bidang ini adalah `N/A` (tidak berlaku).

Mendapatkan laporan kredensial (konsol)

Anda dapat menggunakan file AWS Management Console untuk mengunduh laporan kredensi sebagai file values () CSV yang dipisahkan koma.

Untuk mengunduh laporan kredensial (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Laporan kredensial.
3. Pilih Unduh Laporan.

Mendapatkan laporan kredensial (AWS CLI)

Untuk mengunduh laporan kredensial (AWS CLI)

1. Menghasilkan laporan kredensial. AWS menyimpan satu laporan. Jika ada laporan, membuat laporan kredensial akan menimpa laporan sebelumnya. `aws iam generate-credential-report`
2. Melihat laporan terakhir yang dihasilkan: `aws iam get-credential-report`

Mendapatkan laporan kredensi ()AWS API

Untuk mengunduh laporan kredensial (AWS API)

1. Menghasilkan laporan kredensial. AWS menyimpan satu laporan. Jika ada laporan, membuat laporan kredensial akan menimpa laporan sebelumnya. `GenerateCredentialReport`
2. Melihat laporan terakhir yang dihasilkan: `GetCredentialReport`

IAMkredensial untuk: Kredensial CodeCommit Git, SSH kunci, dan kunci akses AWS

CodeCommit adalah layanan kontrol versi terkelola yang menampung repositori Git pribadi di cloud. AWS Untuk menggunakan CodeCommit, Anda mengkonfigurasi klien Git Anda untuk berkomunikasi dengan CodeCommit repositori. Sebagai bagian dari konfigurasi ini, Anda memberikan IAM kredensil yang CodeCommit dapat digunakan untuk mengautentikasi Anda. IAMmendukung CodeCommit dengan tiga jenis kredensil:

- Kredensial Git, pasangan nama pengguna dan kata sandi yang IAM dihasilkan yang dapat Anda gunakan untuk berkomunikasi dengan CodeCommit repositori. HTTPS
- SSHkeys, sebuah key pair public-private yang dihasilkan secara lokal yang dapat Anda kaitkan dengan IAM pengguna Anda untuk berkomunikasi dengan CodeCommit repositori. SSH
- AWS kunci akses, yang dapat Anda gunakan dengan pembantu kredenal yang disertakan dengan AWS CLI untuk berkomunikasi dengan CodeCommit repositori. HTTPS

> **Note**
>
> Anda tidak dapat menggunakan SSH kunci atau kredensi Git untuk mengakses repositori di akun lain. AWS Untuk mempelajari cara mengonfigurasi akses ke CodeCommit repositori bagi IAM pengguna dan grup di tempat lain Akun AWS, lihat Mengonfigurasi akses lintas akun ke AWS CodeCommit repositori menggunakan peran di Panduan Pengguna.AWS CodeCommit

Lihat bagian berikut untuk informasi selengkapnya tentang setiap opsi.

Gunakan kredensi Git dan HTTPS dengan CodeCommit (disarankan)

Dengan kredensi Git, Anda menghasilkan nama pengguna statis dan pasangan kata sandi untuk IAM pengguna Anda, dan kemudian menggunakan kredensialnya untuk koneksi. HTTPS Anda juga dapat menggunakan kredensil ini dengan alat pihak ketiga atau lingkungan pengembangan terintegrasi (IDE) yang mendukung kredensial Git statis.

Karena kredensial ini bersifat universal untuk semua sistem operasi yang didukung dan sistem manajemen kredensial yang paling kompatibel, lingkungan pengembangan, dan alat pengembangan perangkat lunak lainnya, ini adalah metode yang direkomendasikan. Anda dapat mengatur ulang

kata sandi untuk kredensial Git kapan saja. Anda juga dapat membuat kredensial tidak aktif atau menghapusnya jika Anda tidak lagi membutuhkannya.

> **Note**
>
> Anda tidak dapat memilih nama pengguna atau kata sandi Anda sendiri untuk kredensial Git. IAMmenghasilkan kredensi ini bagi Anda untuk membantu memastikan mereka memenuhi standar keamanan AWS dan mengamankan repositori di. CodeCommit Anda dapat mengunduh kredensialnya hanya sekali, pada saat mereka dihasilkan. Pastikan Anda menyimpan kredensial di lokasi yang aman. Jika perlu, Anda dapat mengatur ulang kata sandi kapan saja, tetapi melakukan hal tersebut dapat membatalkan koneksi yang dikonfigurasi dengan kata sandi lama. Anda harus mengonfigurasi ulang koneksi agar dapat menggunakan kata sandi baru sebelum terhubung.

Lihat topik berikut untuk informasi selengkapnya:

- Untuk membuat pengguna IAM, lihat Buat IAM pengguna di Akun AWS.
- Untuk membuat dan menggunakan kredensial Git dengan CodeCommit, lihat Untuk HTTPS Pengguna yang Menggunakan Kredensial Git di Panduan Pengguna.AWS CodeCommit

> **Note**
>
> Mengubah nama IAM pengguna setelah membuat kredensil Git tidak mengubah nama pengguna dari kredensi Git. Nama pengguna dan kata sandi tetap sama dan masih valid.

Untuk memperbarui kredensi khusus layanan

1. Buat kredensial spesifik layanan kedua yang diatur sebagai tambahan yang saat ini digunakan.
2. Perbarui semua aplikasi Anda untuk menggunakan set kredensial baru dan validasi bahwa aplikasi berfungsi dengan baik.
3. Ubah status kredensial asli menjadi “Inactive”.
4. Pastikan semua aplikasi Anda masih bekerja.
5. Hapus kredensial khusus layanan tidak aktif.

Gunakan SSH kunci dan SSH dengan CodeCommit

Dengan SSH koneksi, Anda membuat file kunci publik dan pribadi di mesin lokal Anda yang Git dan CodeCommit gunakan untuk SSH otentikasi. Anda mengaitkan kunci publik dengan IAM pengguna Anda dan menyimpan kunci pribadi di komputer lokal Anda. Lihat topik berikut untuk informasi selengkapnya:

- Untuk membuat pengguna IAM, lihat Buat IAM pengguna di Akun AWS.
- Untuk membuat kunci SSH publik dan mengaitkannya dengan IAM pengguna, lihat Untuk SSH Koneksi di Linux, macOS, atau Unix atau lihat Untuk SSH Koneksi di Windows di Panduan Pengguna.AWS CodeCommit

> **Note**
>
> Kunci publik harus dikodekan dalam format atau format ssh-rsa. PEM Panjang-bit minimum kunci publik adalah 2048 bit, dan panjang maksimum adalah 16384 bit. Ini terpisah dari ukuran file yang Anda unggah. Misalnya, Anda dapat menghasilkan kunci 2048-bit, dan PEM file yang dihasilkan panjangnya 1679 byte. Jika Anda memberikan kunci publik Anda dalam format atau ukuran lain, Anda akan melihat pesan kesalahan yang menyatakan bahwa format kunci tidak valid.

Gunakan HTTPS dengan pembantu AWS CLI kredensi dan CodeCommit

Sebagai alternatif untuk HTTPS koneksi dengan kredensi Git, Anda dapat mengizinkan Git untuk menggunakan versi IAM kredensi pengguna atau EC2 peran instans Amazon yang ditandatangani secara kriptografis kapan pun Git perlu melakukan autentikasi untuk berinteraksi dengan repositori. AWS CodeCommit Ini adalah satu-satunya metode koneksi untuk CodeCommit repositori yang tidak memerlukan pengguna. IAM Ini juga merupakan satu-satunya metode yang bekerja dengan akses federasi dan kredensial sementara. Lihat topik berikut untuk informasi selengkapnya:

- Untuk mempelajari lebih banyak membahas akses federasi, lihat Penyedia dan federasi identitas dan Akses ke pengguna yang diautentikasi secara eksternal (federasi identitas).
- Untuk mempelajari lebih lanjut tentang kredensil sementara, lihat Kredensi keamanan sementara di IAM dan Akses Sementara ke CodeCommit Repositori.

Pembantu AWS CLI kredenal tidak kompatibel dengan sistem pembantu kredenal lainnya, seperti Akses Rantai Kunci atau Manajemen Kredenal Windows. Ada pertimbangan konfigurasi tambahan saat Anda mengonfigurasi HTTPS koneksi dengan pembantu kredenal. Untuk informasi selengkapnya, lihat Untuk HTTPS Koneksi di Linux, macOS, atau Unix dengan AWS CLI Credential Helper atau HTTPSKoneksi di Windows dengan AWS CLI Credential Helper di Panduan Pengguna.AWS CodeCommit

Kelola sertifikat server di IAM

Untuk mengaktifkan HTTPS koneksi ke situs web atau aplikasi Anda AWS, Anda memerlukan sertifikat SSL TLS /server. Untuk sertifikat di Wilayah yang didukung oleh AWS Certificate Manager (ACM), sebaiknya Anda gunakan ACM untuk menyediakan, mengelola, dan menerapkan sertifikat server Anda. Di Wilayah yang tidak didukung, Anda harus menggunakan IAM sebagai manajer sertifikat. Untuk mempelajari Wilayah mana yang ACM mendukung, lihat AWS Certificate Manager titik akhir dan kuota di. Referensi Umum AWS

> **Important**
>
> ACMadalah alat yang disukai untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Dengan ACM Anda dapat meminta sertifikat atau menyebarkan sertifikat yang ada ACM atau eksternal ke AWS sumber daya. Sertifikat yang ACM disediakan oleh gratis dan diperpanjang secara otomatis. Di Wilayah yang didukung, Anda dapat menggunakan ACM untuk mengelola sertifikat server dari konsol atau secara terprogram. Untuk informasi selengkapnya tentang penggunaanACM, lihat Panduan AWS Certificate Manager Pengguna. Untuk informasi selengkapnya tentang meminta ACM sertifikat, lihat Meminta Sertifikat Publik atau Meminta Sertifikat Pribadi di Panduan AWS Certificate Manager Pengguna. Untuk informasi selengkapnya tentang mengimpor sertifikat pihak ketigaACM, lihat Mengimpor Sertifikat di AWS Certificate Manager Panduan Pengguna.

Gunakan IAM sebagai manajer sertifikat hanya jika Anda harus mendukung HTTPS koneksi di Wilayah yang tidak didukung oleh ACM. IAMmengenkripsi kunci pribadi Anda dengan aman dan menyimpan versi terenkripsi dalam penyimpanan sertifikat. IAM SSL IAMmendukung penerapan sertifikat server di semua Wilayah, tetapi Anda harus mendapatkan sertifikat Anda dari penyedia eksternal untuk digunakan. AWS Anda tidak dapat mengunggah ACM sertifikat keIAM. Selain itu, Anda tidak dapat mengelola sertifikat dari IAM Konsol.

Untuk informasi selengkapnya tentang mengunggah sertifikat pihak ketigaIAM, lihat topik berikut.

Topik

Unggah sertifikat server (AWS API)

Untuk mengunggah sertifikat serverIAM, Anda harus memberikan sertifikat dan kunci privat yang cocok. Jika sertifikat tidak ditandatangani sendiri, Anda juga harus memberikan rantai sertifikat. (Anda tidak memerlukan rantai sertifikat saat mengunggah sertifikat yang ditandatangani sendiri.) Sebelum Anda mengunggah sertifikat, pastikan bahwa Anda memiliki semua item ini dan bahwa mereka memenuhi kriteria berikut:

- Sertifikat harus valid pada saat diunggah. Anda tidak dapat mengunggah sertifikat sebelum periode validitas dimulai (tanggal `NotBefore` sertifikat) atau setelah kedaluwarsa (tanggal `NotAfter` sertifikat).
- Kunci pribadi harus tidak dienkripsi. Anda tidak dapat mengunggah kunci pribadi yang dilindungi oleh kata sandi atau frasa sandi. Untuk membantu mendekripsi kunci pribadi terenkripsi, lihat Memecahkan masalah sertifikat server.
- Sertifikat, kunci pribadi, dan rantai sertifikat semuanya harus PEM dikodekan. Untuk bantuan mengonversi item ini ke PEM format, lihatMemecahkan masalah sertifikat server.

Untuk menggunakan IAMAPIuntuk mengunggah sertifikat, kirim UploadServerCertificatepermintaan. Contoh berikut ini menunjukkan cara melakukan dengan AWS Command Line Interface (AWS CLI). Contoh tersebut mengasumsikan sebagai berikut:

- Sertifikat PEM -encoded disimpan dalam file bernama. `Certificate.pem`
- Rantai sertifikat PEM -encoded disimpan dalam file bernama. `CertificateChain.pem`
- Kunci pribadi yang PEM dienkode dan tidak terenkripsi disimpan dalam file bernama. `PrivateKey.pem`
- (Opsional) Anda ingin menandai sertifikat server dengan pasangan kunci-nilai. Misalnya, Anda dapat menambahkan tombol tanda `Department` dan nilai tanda`Engineering` untuk membantu Anda mengidentifikasi dan mengatur sertifikat Anda.

Untuk menggunakan perintah contoh berikut, ganti nama file berikut dengan nama Anda sendiri. Ganti *ExampleCertificate* dengan nama untuk sertifikat unggahan Anda. Jika Anda ingin menandai sertifikat, ganti *ExampleKey* and *ExampleValue* tandai pasangan nilai kunci dengan nilai Anda sendiri. Ketik perintah pada satu garis kontinu. Contoh berikut mencakup jeda baris dan ruang tambahan untuk memudahkan Anda membaca.

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
                                  --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Ketika perintah sebelumnya berhasil, ia mengembalikan metadata tentang sertifikat yang diunggah, termasuk Amazon Resource Name (ARN), nama ramahnya, pengenal (ID), tanggal kedaluwarsa, tag, dan banyak lagi.

> **Note**
>
> Jika Anda mengunggah sertifikat server untuk digunakan dengan Amazon CloudFront, Anda harus menentukan jalur menggunakan `--path` opsi. Jalur harus dimulai dengan `/cloudfront` dan harus menyertakan garis miring yang menelusur (misalnya, `/cloudfront/test/`).

Untuk menggunakan AWS Tools for Windows PowerShell untuk mengunggah sertifikat, gunakan Publish- IAMServerCertificate.

AWS APIoperasi untuk sertifikat server

Gunakan perintah berikut untuk melihat, menandai, mengganti nama, dan menghapus sertifikat server.

- Gunakan GetServerCertificateuntuk mengambil sertifikat. Permintaan ini mengembalikan sertifikat, rantai sertifikat (jika ada yang diunggah), dan metadata tentang sertifikat.

> **Note**
>
> Anda tidak dapat mengunduh atau mengambil kunci pribadi IAM setelah Anda mengunggahnya.

- Gunakan Get- IAMServerCertificate untuk mengambil sertifikat.
- Gunakan ListServerCertificatesuntuk mencantumkan sertifikat server yang diunggah. Permintaan mengembalikan daftar yang berisi metadata tentang setiap sertifikat.
- Gunakan Get- IAMServerCertificates untuk mencantumkan sertifikat server yang diunggah.
- Gunakan TagServerCertificateuntuk menandai sertifikat server yang ada.
- Gunakan UntagServerCertificateuntuk menghapus tag sertifikat server.
- Gunakan UpdateServerCertificateuntuk mengganti nama sertifikat server atau memperbarui jalurnya.

  Contoh berikut ini menunjukkan cara melakukan ini dengan AWS CLI.

  Untuk menggunakan perintah contoh berikut, ganti nama sertifikat lama dan baru serta jalur sertifikat, lalu ketik perintah pada satu baris berkelanjutan. Contoh berikut mencakup jeda baris dan ruang tambahan untuk memudahkan Anda membaca.

  ```
  aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                    --new-server-certificate-
  name CloudFrontCertificate
                                    --new-path /cloudfront/
  ```

  Untuk menggunakan AWS Tools for Windows PowerShell untuk mengganti nama sertifikat server atau memperbarui jalurnya, gunakan Update- IAMServerCertificate.
- Gunakan DeleteServerCertificateuntuk menghapus sertifikat server.

  Untuk menggunakan AWS Tools for Windows PowerShell untuk menghapus sertifikat server, gunakan Hapus- IAMServerCertificate.

Memecahkan masalah sertifikat server

Sebelum Anda dapat mengunggah sertifikatIAM, Anda harus memastikan bahwa sertifikat, kunci pribadi, dan rantai sertifikat semuanya PEM dikodekan. Anda juga harus memastikan bahwa kunci pribadi tidak terenkripsi. Lihat contoh-contoh berikut.

Example Contoh sertifikat yang PEM dikodekan

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example Contoh kunci pribadi PEM yang dikodekan dan tidak terenkripsi

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example Contoh rantai sertifikat PEM yang dikodekan

Rantai sertifikat berisi satu atau beberapa sertifikat. Anda dapat menggunakan editor teks, perintah penyalinan di Windows, atau perintah cat Linux untuk menggabungkan file sertifikat Anda ke dalam rantai. Jika Anda menyertakan beberapa sertifikat, setiap sertifikat harus mensertifikasi sertifikat sebelumnya. Anda mencapai ini dengan menggabungkan sertifikat, termasuk sertifikat CA akar terakhir.

Contoh berikut berisi tiga sertifikat, tetapi rantai sertifikat Anda mungkin memuat lebih banyak atau lebih sedikit sertifikat.

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Jika item ini tidak dalam format yang tepat untuk diunggahIAM, Anda dapat menggunakan Buka SSL untuk mengonversinya ke format yang tepat.

Untuk mengonversi sertifikat atau rantai sertifikat dari DER ke PEM

Gunakan perintah Open SSL x509, seperti pada contoh berikut. Dalam contoh perintah berikut, ganti *Certificate.der* dengan nama file yang berisi sertifikat DER -encoded Anda. Ganti *Certificate.pem* dengan nama yang disukai dari file output untuk berisi sertifikat PEM -encoded.

```
openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

Untuk mengonversi kunci pribadi dari DER ke PEM

Gunakan perintah Open SSL rsa, seperti pada contoh berikut. Dalam contoh perintah berikut, ganti *PrivateKey.der* dengan nama file yang berisi kunci pribadi DER -encoded Anda. Ganti *PrivateKey.pem* dengan nama yang disukai dari file output untuk berisi kunci pribadi PEM -encoded.

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

Untuk mendekripsi kunci pribadi terenkripsi (hapus kata sandi atau frasa sandi)

Gunakan perintah Open SSL rsa, seperti pada contoh berikut. Untuk menggunakan perintah contoh berikut, ganti *EncryptedPrivateKey.pem* dengan nama file yang berisi kunci pribadi Anda yang dienkripsi. Ganti *PrivateKey.pem* dengan nama yang disukai dari file output untuk berisi kunci pribadi tak PEM terenkripsi -encode.

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

Untuk mengonversi bundel sertifikat dari PKCS #12 (PFX) ke PEM

Gunakan perintah Open SSL pkcs12, seperti pada contoh berikut. Dalam contoh perintah berikut, ganti *CertificateBundle.p12* dengan nama file yang berisi bundel sertifikat PKCS #12 -encoded Anda. Ganti *CertificateBundle.pem* dengan nama yang disukai dari file output untuk berisi bundel sertifikat PEM -encoded.

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

Untuk mengonversi bundel sertifikat dari PKCS #7 ke PEM

Gunakan perintah Open SSL pkcs7, seperti pada contoh berikut. Dalam contoh perintah berikut, ganti *CertificateBundle.p7b* dengan nama file yang berisi bundel sertifikat PKCS #7 - encoded Anda. Ganti *CertificateBundle.pem* dengan nama yang disukai dari file output untuk berisi bundel sertifikat PEM -encoded.

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

Grup pengguna IAM

Grup IAM pengguna adalah kumpulan IAM pengguna. Grup pengguna memungkinkan Anda menentukan izin untuk beberapa pengguna, yang dapat memudahkan pengelolaan izin bagi pengguna tersebut. Misalnya, Anda dapat memiliki grup pengguna bernama Admin dan memberikan izin administrator khas grup pengguna tersebut. Setiap pengguna dalam grup pengguna tersebut secara otomatis memiliki izin grup Admin. Jika pengguna baru bergabung dengan organisasi Anda dan memerlukan hak administrator, Anda dapat menetapkan izin yang sesuai dengan menambahkan pengguna ke grup pengguna Admin. Jika seseorang mengubah pekerjaan di organisasi Anda, alih-alih mengedit izin pengguna tersebut, Anda dapat menghapusnya dari IAM grup lama dan menambahkannya ke IAM grup baru yang sesuai.

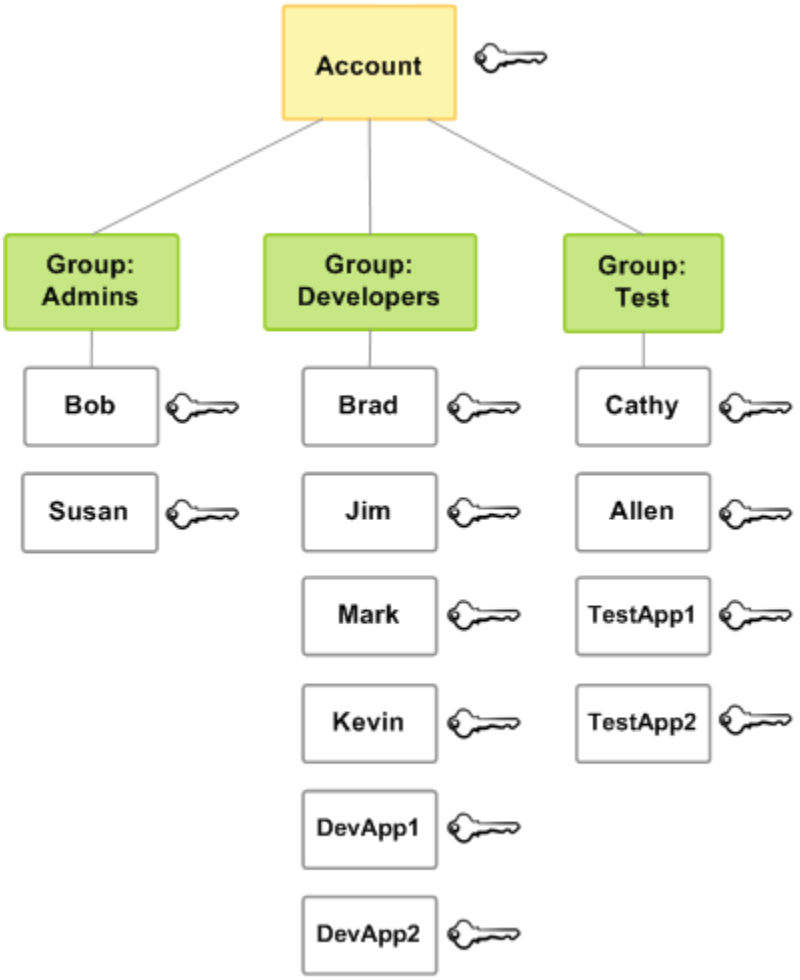
Anda dapat melampirkan kebijakan berbasis identitas ke grup pengguna sehingga semua pengguna dalam grup pengguna menerima izin kebijakan tersebut. Anda tidak dapat mengidentifikasi grup pengguna sebagai `Principal` dalam kebijakan (seperti kebijakan berbasis sumber daya) karena grup terkait dengan izin, bukan autentikasi, dan prinsipal adalah entitas yang diautentikasi. IAM Untuk informasi lebih lanjut tentang jenis kebijakan, lihat Kebijakan berbasis identitas dan kebijakan berbasis sumber daya.

Berikut adalah beberapa karakteristik penting dari IAM kelompok:

- Sebuah grup pengguna dapat berisi banyak pengguna, dan pengguna dapat menjadi bagian dari beberapa grup pengguna.
- Grup pengguna tidak dapat disarangkan; mereka hanya dapat berisi pengguna, bukan IAM grup lain.

- Tidak ada grup pengguna default yang secara otomatis menyertakan semua pengguna di Akun AWS. Jika Anda ingin memiliki grup pengguna seperti itu, Anda harus membuatnya dan menetapkan setiap pengguna baru ke grup tersebut.
- Jumlah dan ukuran IAM sumber daya dalam Akun AWS, seperti jumlah grup, dan jumlah grup yang pengguna dapat menjadi anggota, terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Diagram berikut menunjukkan contoh sederhana dari perusahaan kecil. Pemilik perusahaan membuat grup penggna `Admins` bagi pengguna untuk membuat dan mengelola pengguna lain seiring pertumbuhan perusahaan. Parametergrup pengguna `Admins` membuat grup pengguna `Developers` dan grup pengguna `Test`. Masing-masing IAM grup ini terdiri dari pengguna (manusia dan aplikasi) yang berinteraksi dengan AWS (Jim, Brad, DevApp 1, dan sebagainya). Setiap pengguna memiliki satu set kredensial keamanan. Dalam contoh ini, setiap pengguna termasuk dalam satu grup pengguna. Namun, pengguna dapat menjadi bagian dari beberapa IAM grup.



Buat grup IAM pengguna

> **Note**
>
> Sebagai praktik terbaik, kami menyarankan Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara. Jika Anda mengikuti praktik terbaik, Anda tidak mengelola IAM pengguna dan grup. Sebaliknya, pengguna dan grup Anda dikelola di luar AWS dan dapat mengakses AWS sumber daya sebagai identitas federasi. Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses AWS layanan dengan menggunakan kredensi yang disediakan melalui sumber identitas. Identitas federasi menggunakan grup yang ditentukan oleh penyedia identitas mereka. Jika Anda menggunakan AWS IAM Identity Center, lihat Mengelola identitas di Pusat IAM Identitas di Panduan AWS IAM Identity Center Pengguna untuk informasi tentang membuat pengguna dan grup di Pusat IAM Identitas.

Untuk mengatur grup pengguna, Anda perlu membuat grup. Kemudian berikan izin grup berdasarkan jenis pekerjaan yang Anda harapkan dilakukan pengguna dalam grup. Terakhir, tambahkan pengguna ke grup.

Untuk informasi tentang izin yang Anda butuhkan untuk membuat grup pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Untuk membuat grup IAM pengguna dan melampirkan kebijakan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Dalam panel navigasi, pilih User groups (Grup pengguna) lalu pilih Create New Group (Buat Grup Baru).
3. Untuk User grup name (Nama grup), ketik nama grup.

   > **Note**
   >
   > Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota. Nama grup dapat berupa kombinasi hingga 128 huruf, digit, dan karakter ini: plus (+), sama (=), koma (,), titik (.), pada tanda

(@), garis bawah (_), dan tanda hubung (-). Nama harus unik dalam akun. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat grup dengan nama keduanya **`ADMINS`** dan **`admins`**.

4. Dalam daftar pengguna, pilih kotak centang untuk setiap pengguna yang ingin Anda tambahkan ke grup.
5. Dalam daftar kebijakan, centang kotak untuk setiap kebijakan yang ingin Anda terapkan pada semua anggota grup.
6. Pilih Buat grup.

Untuk membuat grup IAM pengguna (AWS CLI atau AWS API)

Gunakan salah satu langkah berikut:

- AWS CLI: aws iam create-group
- AWS API: CreateGroup

Lihat IAM grup

Anda dapat membuat daftar semua IAM grup di akun Anda, daftar pengguna dalam grup pengguna, dan daftar IAM grup milik pengguna. Jika Anda menggunakan AWS CLI or AWS API, Anda dapat mencantumkan semua IAM grup dengan awalan jalur tertentu.

Untuk mencantumkan semua IAM grup di akun Anda

Lakukan salah satu langkah berikut ini:

- AWS Management Console: Di panel navigasi, pilih Grup pengguna.
- AWS CLI: aws iam list-groups
- AWS API: ListGroups

Untuk mencantumkan pengguna dalam grup pengguna tertentu

Lakukan langkah-langkah berikut:

- AWS Management Console: Di panel navigasi, pilih Grup pengguna, pilih nama grup, lalu pilih tab Pengguna.

- AWS CLI: aws iam get-group
- AWS API: GetGroup

Untuk membuat daftar semua IAM grup tempat pengguna berada

Lakukan salah satu langkah berikut ini:

- AWS Management Console: Di panel navigasi, pilih Pengguna, pilih nama pengguna, lalu pilih tab Grup.
- AWS CLI: aws iam list-groups-for -pengguna
- AWS API: ListGroupsForUser

Mengedit pengguna dalam IAM IAM grup

Gunakan IAM grup untuk menerapkan kebijakan izin yang sama di beberapa pengguna sekaligus. Anda kemudian dapat menambahkan pengguna ke atau menghapus pengguna dari grup IAM pengguna. Hal ini berguna saat orang-orang memasuki dan meninggalkan organisasi Anda.

Lihat akses kebijakan

Sebelum Anda mengubah izin untuk suatu kebijakan, Anda harus meninjau aktivitas tingkat layanannya yang terbaru. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya tentang melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menambahkan atau menghapus pengguna dalam grup pengguna (konsol)

Anda dapat menggunakan AWS Management Console untuk menambah atau menghapus pengguna dari grup pengguna.

Untuk menambahkan pengguna ke grup IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna lalu pilih nama grup.
3. Pilih tab Pengguna dan kemudian pilih Tambahkan pengguna. Pilih kotak centang di samping pengguna yang ingin Anda tambahkan.

4. Pilih Tambahkan pengguna.

Untuk menghapus pengguna dari IAM grup (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna lalu pilih nama grup.
3. Pilih tab Pengguna. Pilih kotak centang di samping pengguna yang ingin Anda hapus, lalu pilih Hapus pengguna.

Menambahkan atau menghapus pengguna dalam grup pengguna (AWS CLI)

Anda dapat menggunakan AWS CLI untuk menambah atau menghapus pengguna dari grup pengguna.

Untuk menambahkan pengguna ke grup IAM pengguna (AWS CLI)

- Gunakan perintah berikut ini.
  - aws iam add-user-to -grup

Untuk menghapus pengguna dari grup IAM pengguna (AWS CLI)

- Gunakan perintah berikut ini.
  - aws iam remove-user-from -grup

Menambah atau menghapus pengguna dalam grup pengguna (AWS API)

Anda dapat menggunakan AWS API untuk menambah atau menghapus pengguna dalam grup pengguna.

Untuk menambahkan pengguna ke IAM grup (AWS API)

- Selesaikan operasi berikut:
  - AddUserToGroup

Untuk menghapus pengguna dari grup IAM pengguna (AWS API)

- Selesaikan operasi berikut:
  - RemoveUserFromGroup

Melampirkan kebijakan ke grup IAM pengguna

Anda dapat melampirkan kebijakan AWS terkelola —yaitu kebijakan pratulis yang disediakan oleh AWS—ke grup pengguna, seperti yang dijelaskan dalam langkah-langkah berikut. Untuk melampirkan kebijakan yang dikelola pelanggan—yaitu, izin khusus dengan kebijakan yang Anda buat—Anda harus membuat kebijakan terlebih dahulu. Untuk informasi tentang membuat kebijakan yang dikelola pelanggan, lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan.

Untuk informasi lebih lanjut tentang izin dan kebijakan, lihat Manajemen akses untuk AWS sumber daya.

Untuk melampirkan kebijakan ke grup pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna lalu pilih nama grup.
3. Pilih tab Izin.
4. Pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.
5. Kebijakan yang saat ini terlampir ke grup pengguna ditampilkan dalam daftar Kebijakan izin saat ini. Dalam daftar Kebijakan izin lainnya, pilih kotak centang di samping nama kebijakan yang akan dilampirkan. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan menurut jenis dan nama kebijakan.
6. Pilih kebijakan yang ingin dilampirkan ke grup IAM pengguna, lalu pilih Lampirkan kebijakan.

Untuk melampirkan kebijakan ke grup pengguna (AWS CLI atau AWS API)

Lakukan salah satu dari langkah berikut:

- AWS CLI: aws iam attach-group-policy
- AWS API: AttachGroupPolicy

Ganti nama grup IAM pengguna

Saat Anda mengubah nama atau alur grup pengguna, hal berikut terjadi:

- Kebijakan apa pun yang terlampir pada grup pengguna akan tetap melekat ke grup dengan nama baru.
- Grup pengguna mempertahankan semua penggunanya di bawah nama baru.
- ID unik untuk grup pengguna tetap sama. Untuk informasi lebih lanjut tentang unikIDs, lihatPengidentifikasi unik.

IAMtidak secara otomatis memperbarui kebijakan yang merujuk ke grup pengguna sebagai sumber daya untuk menggunakan nama baru. Oleh karena itu, Anda harus berhati-hati ketika mengubah nama grup pengguna. Sebelum mengubah nama grup pengguna, Anda harus memeriksa semua kebijakan secara manual untuk menemukan kebijakan yang menyebutkan nama grup pengguna tersebut. Misalnya, katakanlah Bob adalah manajer bagian pengujian dari organisasi. Bob memiliki kebijakan yang dilampirkan ke entitas IAM penggunanya yang memungkinkan dia menambah dan menghapus pengguna dari grup pengguna Uji. Jika administrator mengubah nama grup pengguna (atau mengubah jalur grup), administrator juga harus memperbarui kebijakan yang dilampirkan ke Bob untuk menggunakan nama atau jalur baru. Jika tidak, Bob tidak akan dapat menambahkan dan menghapus pengguna dari grup pengguna.

Untuk menemukan kebijakan yang merujuk suatu grup pengguna sebagai sumber daya:

1. Dari panel navigasi IAM konsol, pilih Kebijakan.
2. Urutkan menurut kolom Jenis untuk kebijakan kustom yang Dikelola pelanggan.
3. Pilih nama kebijakan yang ingin diedit.
4. Pilih tab Izin, lalu pilih Ringkasan.
5. Pilih IAMdari daftar layanan, jika ada.
6. Cari nama grup pengguna Anda di kolom Sumber Daya.
7. Pilih Edit untuk mengubah nama grup pengguna Anda dalam kebijakan.

Untuk mengubah nama grup IAM pengguna

Lakukan salah satu langkah berikut ini:

- AWS Management Console: Di panel navigasi, pilih Grup pengguna lalu pilih nama grup. Pilih Edit. Ketik nama grup pengguna baru lalu pilih Simpan perubahan.
- AWS CLI: aws iam update-group
- AWS API: UpdateGroup

Menghapus grup pengguna IAM

Saat Anda menghapus grup pengguna di AWS Management Console, konsol secara otomatis menghapus semua anggota grup, melepaskan semua kebijakan terkelola yang dilampirkan, dan menghapus semua kebijakan sebaris. Namun, karena IAM tidak secara otomatis menghapus kebijakan yang merujuk ke grup pengguna sebagai sumber daya, Anda harus berhati-hati saat menghapus grup pengguna. Sebelum menghapus grup pengguna, Anda harus memeriksa semua kebijakan Anda secara manual untuk menemukan kebijakan yang menyebutkan nama grup tersebut. Misalnya, John, manajer Tim Uji, memiliki kebijakan yang dilampirkan ke entitas IAM penggunanya yang memungkinkannya menambah dan menghapus pengguna dari grup pengguna Uji. Jika administrator menghapus grup tersebut, administrator juga harus menghapus kebijakan yang terlampir pada John. Jika tidak, jika administrator membuat ulang grup yang dihapus dan memberinya nama yang sama, izin John tetap ada, bahkan jika dia meninggalkan Tim Uji.

Untuk menemukan kebijakan yang merujuk suatu grup pengguna sebagai sumber daya

1. Dari panel navigasi IAM konsol, pilih Kebijakan.
2. Urutkan menurut kolom Jenis untuk kebijakan kustom yang Dikelola pelanggan.
3. Pilih nama kebijakan yang ingin dihapus.
4. Pilih tab Izin, lalu pilih Ringkasan.
5. Pilih IAMdari daftar layanan, jika ada.
6. Cari nama grup pengguna Anda di kolom Sumber Daya.
7. Pilih Hapus untuk menghapus kebijakan.
8. Ketik nama kebijakan untuk mengonfirmasi penghapusan kebijakan dan pilih Hapus.

Sebaliknya, ketika Anda menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk menghapus grup pengguna, Anda harus terlebih dahulu menghapus pengguna dalam grup. Kemudian hapus kebijakan inline apa pun yang disematkan dalam grup pengguna. Berikutnya, lepaskan kebijakan terkelola yang melekat pada grup. Hanya dengan begitu Anda dapat menghapus grup pengguna itu sendiri.

Menghapus grup pengguna IAM (konsol)

Anda dapat menghapus grup IAM pengguna dari file AWS Management Console.

Untuk menghapus grup IAM pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih User groups (Grup pengguna).
3. Dalam daftar IAM grup, pilih kotak centang di sebelah nama IAM grup yang akan dihapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar IAM grup berdasarkan jenis, izin, dan nama grup pengguna.
4. Pilih Hapus.
5. Dalam kotak konfirmasi, jika Anda ingin menghapus satu grup pengguna, ketik nama grup pengguna dan pilih Hapus. Jika Anda ingin menghapus beberapa grup pengguna, ketikkan jumlah IAM grup yang akan dihapus diikuti **`user groups`** dan pilih Hapus. Misalnya, jika Anda menghapus tiga IAM grup, ketik**`3 IAM groups`**.

Menghapus grup pengguna IAM (AWS CLI)

Anda dapat menghapus grup IAM pengguna dari file AWS CLI.

Untuk menghapus grup pengguna IAM (AWS CLI)

1. Menghapus semua pengguna dari grup pengguna.
   - aws iam get-group (untuk mendapatkan daftar pengguna di grup pengguna), dan aws iam remove-user-from -group (untuk menghapus pengguna dari grup pengguna)
2. Menghapus semua kebijakan inline yang disertakan dalam grup pengguna.
   - aws iam list-group-policies (untuk mendapatkan daftar kebijakan sebaris grup pengguna), dan aws iam delete-group-policy (untuk menghapus kebijakan sebaris grup pengguna)
3. Melepaskan semua kebijakan terkelola yang terlampir pada grup pengguna.
   - aws iam list-attached-group -policies (untuk mendapatkan daftar kebijakan terkelola yang dilampirkan ke grup pengguna), dan aws iam detach-group-policy (untuk melepaskan kebijakan terkelola dari grup pengguna)
4. Menghapus grup pengguna.

- aws iam hapus-grup

Menghapus grup pengguna IAM (AWS API)

Anda dapat menggunakan AWS API untuk menghapus grup IAM pengguna.

Untuk menghapus grup IAM pengguna (AWS API)

1. Menghapus semua pengguna dari grup pengguna.
   - GetGroup(untuk mendapatkan daftar pengguna di grup pengguna) dan RemoveUserFromGroup(untuk menghapus pengguna dari grup pengguna)
2. Menghapus semua kebijakan inline yang disertakan dalam grup pengguna.
   - ListGroupPolicies(untuk mendapatkan daftar kebijakan sebaris grup pengguna) dan DeleteGroupPolicy(untuk menghapus kebijakan sebaris grup pengguna)
3. Melepaskan semua kebijakan terkelola yang terlampir pada grup pengguna.
   - ListAttachedGroupPolicies(untuk mendapatkan daftar kebijakan terkelola yang dilampirkan ke grup pengguna) dan DetachGroupPolicy(untuk melepaskan kebijakan terkelola dari grup pengguna)
4. Menghapus grup pengguna.
   - DeleteGroup

IAMperan

IAMPeran adalah IAM identitas yang dapat Anda buat di akun Anda yang memiliki izin khusus. IAMPeran mirip dengan IAM pengguna, karena itu adalah AWS identitas dengan kebijakan izin yang menentukan apa yang dapat dan tidak dapat dilakukan identitas AWS. Namun, alih-alih secara unik terkait dengan satu orang, peran dimaksudkan untuk menjadi dapat diambil oleh siapa pun yang membutuhkannya. Selain itu, peran tidak memiliki kredensial jangka panjang standar seperti kata sandi atau kunci akses yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, peran tersebut akan memberikan kredensial keamanan sementara untuk sesi peran.

Anda dapat menggunakan peran untuk mendelegasikan akses ke pengguna, aplikasi, atau layanan yang biasanya tidak memiliki akses ke AWS sumber daya Anda. Misalnya, Anda mungkin ingin

memberi pengguna di AWS akun Anda akses ke sumber daya yang biasanya tidak mereka miliki, atau memberikan pengguna dalam satu Akun AWS akses ke sumber daya di akun lain. Atau Anda mungkin ingin mengizinkan aplikasi seluler menggunakan AWS sumber daya, tetapi tidak ingin menyematkan AWS kunci di dalam aplikasi (di mana kunci tersebut sulit diperbarui dan di mana pengguna berpotensi mengekstraknya). Terkadang Anda ingin memberikan AWS akses ke pengguna yang sudah memiliki identitas yang ditentukan di luar AWS, seperti di direktori perusahaan Anda. Atau, Anda mungkin ingin memberikan akses ke akun Anda kepada pihak ketiga sehingga mereka dapat melakukan audit pada sumber daya Anda.

Untuk skenario ini, Anda dapat mendelegasikan akses ke AWS sumber daya menggunakan IAMperan. Bagian ini memperkenalkan peran dan berbagai cara yang dapat Anda gunakan, kapan dan bagaimana memilih di antara pendekatan, dan bagaimana membuat, mengelola, beralih ke (atau mengasumsikan), dan menghapus peran.

> **Note**
>
> Saat pertama kali membuat Akun AWS, tidak ada peran yang dibuat secara default. Saat Anda menambahkan layanan ke akun Anda, mereka dapat menambahkan peran terkait layanan untuk mendukung kasus penggunaannya.
> Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. IAMAdministrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.
> Sebelum Anda dapat menghapus peran terkait layanan, Anda harus terlebih dahulu menghapus sumber daya terkait mereka. Ini melindungi sumber daya Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.
> Untuk informasi tentang layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait-Layanan Pilih Ya dengan tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Topik

- Skenario umum untuk IAM peran
- IAMpenciptaan peran
- IAMmanajemen peran
- Metode untuk mengambil peran

Kapan membuat IAM pengguna (bukan peran)

Kami menyarankan Anda hanya menggunakan IAM pengguna untuk kasus penggunaan yang tidak didukung oleh pengguna federasi. Beberapa kasus penggunaan meliputi yang berikut:

- Beban kerja yang tidak dapat menggunakan IAM peran — Anda dapat menjalankan beban kerja dari lokasi yang perlu diakses. AWS Dalam beberapa situasi, Anda tidak dapat menggunakan IAM peran untuk memberikan kredensi sementara, seperti untuk WordPress plugin. Dalam situasi ini, gunakan kunci akses jangka panjang IAM pengguna untuk beban kerja tersebut untuk mengautentikasi. AWS
- AWS Klien pihak ketiga — Jika Anda menggunakan alat yang tidak mendukung akses dengan Pusat IAM Identitas, seperti AWS klien pihak ketiga atau vendor yang tidak di-host AWS, gunakan kunci akses jangka panjang IAM pengguna.
- AWS CodeCommit akses - Jika Anda menggunakan CodeCommit untuk menyimpan kode Anda, Anda dapat menggunakan IAM pengguna dengan SSH kunci atau kredensi khusus layanan untuk mengautentikasi CodeCommit ke repositori Anda. Kami menyarankan Anda melakukan ini selain menggunakan pengguna di Pusat IAM Identitas untuk otentikasi normal. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud Anda Akun AWS atau ke aplikasi cloud Anda. Untuk memberi pengguna akses ke CodeCommit repositori Anda tanpa mengonfigurasi IAM pengguna, Anda dapat mengonfigurasi utilitas. git-remote-codecommit Untuk informasi lebih lanjut tentang IAM dan CodeCommit, lihatIAMkredensial untuk: Kredensial CodeCommit Git, SSH kunci, dan kunci akses AWS. Untuk informasi selengkapnya tentang mengonfigurasi git-remote-codecommit utilitas, lihat Menghubungkan ke AWS CodeCommit repositori dengan kredensi berputar di Panduan Pengguna.AWS CodeCommit
- Akses Amazon Keyspaces (untuk Apache Cassandra) — Dalam situasi di mana Anda tidak dapat menggunakan pengguna di Pusat IAM Identitas, seperti untuk tujuan pengujian kompatibilitas Cassandra, Anda dapat menggunakan IAM pengguna dengan kredensi khusus layanan untuk mengautentikasi dengan Amazon Keyspaces. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud Anda Akun AWS atau

ke aplikasi cloud Anda. Anda juga dapat terhubung ke Amazon Keyspaces menggunakan kredensil sementara. Untuk informasi selengkapnya, lihat Menggunakan kredensil sementara untuk terhubung ke Amazon Keyspaces menggunakan IAM peran dan plugin SiGv4 di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

- Akses darurat — Dalam situasi di mana Anda tidak dapat mengakses penyedia identitas Anda dan Anda harus mengambil tindakan di dalamnya Akun AWS. Menetapkan IAM pengguna akses darurat dapat menjadi bagian dari rencana ketahanan Anda. Kami menyarankan agar kredensil pengguna darurat dikontrol dan diamankan dengan ketat menggunakan otentikasi multi-faktor (). MFA

Istilah dan konsep peran

Berikut ini beberapa istilah dasar untuk membantu Anda memulai dengan peran.

Peran

IAMIdentitas yang dapat Anda buat di akun Anda yang memiliki izin khusus. IAMPeran memiliki beberapa kesamaan dengan IAM pengguna. Peran dan pengguna adalah identitas AWS dengan kebijakan izin yang menentukan apa yang dapat dan tidak dapat dilakukan oleh identitas tersebut di AWS. Namun, alih-alih secara unik terkait dengan satu orang, peran dimaksudkan untuk menjadi dapat diambil oleh siapa pun yang membutuhkannya. Selain itu, peran tidak memiliki kredensial jangka panjang standar seperti kata sandi atau kunci akses yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, peran tersebut akan memberikan kredensial keamanan sementara untuk sesi peran.

Peran dapat diasumsikan sebagai berikut:

- IAMPengguna dalam hal yang sama Akun AWS atau yang lain Akun AWS
- IAMperan dalam akun yang sama
- Prinsipal layanan, untuk digunakan dengan AWS layanan dan fitur seperti:
  - Layanan yang memungkinkan Anda menjalankan kode pada layanan komputasi, seperti Amazon EC2 atau AWS Lambda
  - Fitur yang melakukan tindakan ke sumber daya Anda atas nama Anda, seperti replikasi objek Amazon S3
  - Layanan yang memberikan kredensil keamanan sementara ke aplikasi Anda yang berjalan di luar AWS, seperti IAM Roles Anywhere atau Amazon Anywhere ECS

- Pengguna eksternal yang diautentikasi oleh layanan penyedia identitas eksternal (IDP) yang kompatibel SAML dengan 2.0 atau OpenID Connect

AWS peran layanan

Peran layanan adalah IAMperan yang diasumsikan layanan untuk melakukan tindakan atas nama Anda. IAMAdministrator dapat membuat, memodifikasi, dan menghapus peran layanan dari dalamIAM. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke Layanan AWS dalam IAMPanduan Pengguna.

AWS peran terkait layanan

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. IAMAdministrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.

> **Note**
>
> Jika Anda sudah menggunakan layanan ketika mulai mendukung peran yang terkait dengan layanan, Anda mungkin menerima email yang mengumumkan peran baru di akun Anda. Dalam hal ini, layanan secara otomatis membuat peran terkait layanan di akun Anda. Anda tidak perlu melakukan tindakan apa pun untuk mendukung peran ini, dan Anda tidak harus menghapusnya secara manual. Untuk informasi selengkapnya, lihat Peran baru muncul di akun AWS saya.

Untuk informasi tentang layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait-Layanan Pilih Ya dengan tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut. Untuk informasi selengkapnya, lihat Buat peran tertaut layanan.

Rantai peran

Role chaining adalah ketika Anda menggunakan peran untuk mengambil peran kedua melalui AWS CLI atauAPI. Misalnya, `RoleA` memiliki izin untuk berasumsi`RoleB`. Anda dapat mengaktifkan User1 untuk berasumsi `RoleA` dengan menggunakan kredensi pengguna jangka panjang mereka dalam operasi. AssumeRole API Ini mengembalikan kredensi `RoleA` jangka pendek. Dengan rantai peran, Anda dapat menggunakan `RoleA` kredensi jangka pendek untuk mengaktifkan User1 berasumsi. `RoleB`

Saat Anda mengambil peran, Anda dapat melewatkan tag sesi dan mengatur tag sebagai transitif. Tag sesi transitif diteruskan ke semua sesi berikutnya dalam rantai peran. Untuk mempelajari lebih lanjut tentang tag sesi, lihat Lulus tag sesi di AWS STS.

Rantai peran membatasi sesi AWS CLI atau AWS API peran Anda hingga maksimal satu jam. Bila Anda menggunakan AssumeRoleAPIoperasi untuk mengambil peran, Anda dapat menentukan durasi sesi peran Anda dengan `DurationSeconds` parameter. Anda dapat menentukan nilai parameter hingga 43200 detik (12 jam), tergantung pada pengaturan durasi sesi maksimum untuk peran Anda. Akan tetapi, jika Anda berperan menggunakan rantai peran dan memberikan nilai parameter `DurationSeconds` lebih dari satu jam, operasi gagal.

Delegasi

Pemberian izin kepada seseorang untuk memungkinkan akses ke sumber daya yang Anda kontrol. Delegasi melibatkan terbentuknya kepercayaan antara dua akun. Yang pertama adalah akun yang memiliki sumber daya (akun terpercaya). Kedua adalah akun yang berisi pengguna yang perlu mengakses sumber daya (akun tepercaya). Akun tepercaya dan dipercaya dapat merupakan salah satu dari berikut ini:

- Akun yang sama.
- Pisahkan akun yang berada di bawah kendali organisasi Anda.
- Dua akun yang dimiliki oleh organisasi yang berbeda.

Untuk mendelegasikan izin untuk mengakses sumber daya, Anda membuat IAM peran di akun kepercayaan yang memiliki dua kebijakan terlampir. Kebijakan izin memberikan izin yang diperlukan pengguna untuk melaksanakan tugas yang diinginkan pada sumber daya. Kebijakan kepercayaan menentukan anggota akun tepercaya mana yang diizinkan untuk menjalankan peran tersebut.

Saat membuat kebijakan kepercayaan, Anda tidak dapat menentukan wildcard (*) sebagai bagian dari dan ARN di elemen utama. Kebijakan terpercaya ini dilampirkan pada peran dalam akun kepercayaan, dan merupakan setengah dari izin. Setengah lainnya adalah kebijakan izin yang melekat pada pengguna dalam akun tepercaya yang memungkinkan pengguna untuk beralih, atau mengambil peran. Seorang pengguna yang mengambil peran secara sementara memberikan izinnya sendiri dan sebaliknya mengambil izin peran tersebut. Saat pengguna keluar, atau berhenti menggunakan peran tersebut, izin pengguna yang asli akan dipulihkan. Parameter tambahan yang disebut ID eksternal membantu memastikan penggunaan peran aman antar akun yang tidak dikontrol oleh organisasi yang sama.

Kebijakan kepercayaan

Dokumen JSON kebijakan di mana Anda mendefinisikan prinsip-prinsip yang Anda percayai untuk mengambil peran. Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya wajib yang melekat pada peran di dalamnya. IAM Prinsipal yang dapat Anda sebutkan dalam kebijakan kepercayaan termasuk pengguna, peran, akun, dan layanan.

Peran untuk akses lintas akun

Peran yang memberikan akses ke sumber daya dalam satu akun ke prinsipal tepercaya dalam akun yang berbeda. Peran adalah cara utama untuk memberikan akses akun silang. Namun, beberapa AWS layanan memungkinkan Anda untuk melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Ini disebut kebijakan berbasis sumber daya, dan Anda dapat menggunakannya untuk memberikan prinsipal di akses lain ke sumber daya. Akun AWS Beberapa sumber daya ini termasuk bucket Amazon Simple Storage Service (S3), kubah S3 Glacier, topik Amazon Simple Notification Service (SNS), dan antrian Amazon Simple Queue Service (). SQS Untuk mempelajari layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM. Untuk informasi selengkapnya tentang kebijakan berbasis sumber daya, lihat Akses sumber daya lintas akun di IAM.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang IAM terminologi yang terkait dengan IAM peran.

- Prinsipal adalah entitas AWS yang dapat melakukan tindakan dan mengakses sumber daya. Prinsipal dapat berupa Pengguna root akun AWS, IAM pengguna, atau peran. Prinsipal yang mewakili identitas suatu AWS layanan adalah prinsipal layanan. Gunakan elemen Principal dalam kebijakan kepercayaan peran untuk menentukan prinsip yang Anda percayai untuk mengambil peran tersebut.

  Untuk informasi lebih lanjut dan contoh prinsip yang dapat Anda izinkan untuk mengambil peran, lihat. AWS JSONelemen kebijakan: Principal
- Federasi identitas menciptakan hubungan kepercayaan antara penyedia identitas eksternal dan AWS. Anda dapat menggunakan penyedia OpenID Connect (OIDC) atau Security Assertion Markup Language (SAML) 2.0 yang sudah ada untuk mengelola siapa saja yang dapat mengakses sumber daya. AWS Saat Anda menggunakan OIDC dan SAML 2.0 untuk mengonfigurasi hubungan kepercayaan antara penyedia identitas eksternal ini dan AWS , pengguna ditetapkan

ke IAM peran. Pengguna juga menerima kredensial sementara yang mengizinkan pengguna mengakses sumber daya AWS Anda.

Untuk informasi selengkapnya tentang pengguna gabungan, lihat Penyedia dan federasi identitas.

- Pengguna federasi adalah identitas yang ada dari AWS Directory Service, direktori pengguna perusahaan Anda, atau penyedia. OIDC Ini dikenal sebagai pengguna federasi. AWS memberikan peran kepada pengguna federasi ketika akses diminta melalui penyedia identitas.

  Untuk informasi selengkapnya tentang pengguna gabungan, lihat Pengguna gabungan dan peran.

- Kebijakan izin adalah kebijakan berbasis identitas yang menentukan tindakan dan sumber daya apa yang dapat digunakan peran tersebut. Dokumen ini ditulis sesuai dengan aturan bahasa IAM kebijakan.

  Untuk informasi selengkapnya, lihat IAMJSONreferensi kebijakan.

- Batas izin adalah fitur lanjutan tempat Anda menggunakan kebijakan untuk membatasi izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas untuk peran. Anda tidak dapat menerapkan batas izin untuk peran yang terkait layanan.

  Untuk informasi selengkapnya, lihat Batas izin untuk entitas IAM.

Masalah confused deputy

Masalah deputi yang bingung adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memaksa entitas yang lebih istimewa untuk melakukan tindakan. Untuk mencegah hal ini, AWS sediakan alat yang membantu Anda melindungi akun Anda jika Anda memberi pihak ketiga (dikenal sebagai lintas akun) atau AWS layanan lain (dikenal sebagai lintas layanan) akses ke sumber daya di akun Anda.

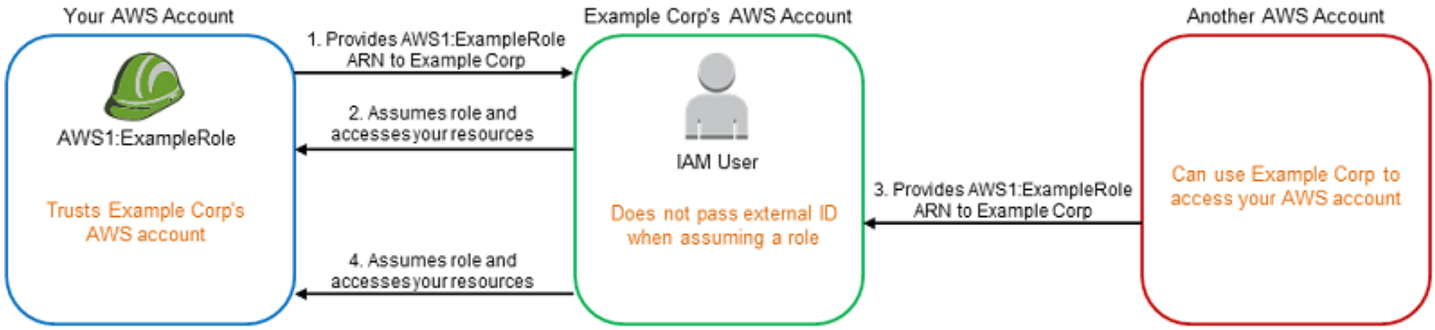
Terkadang, Anda mungkin perlu memberi pihak ketiga akses ke AWS sumber daya Anda (akses delegasi). Misalnya, katakanlah Anda memutuskan untuk menyewa perusahaan pihak ketiga bernama Example Corp untuk memantau Anda Akun AWS dan membantu mengoptimalkan biaya. Untuk melacak pengeluaran harian Anda, Example Corp perlu mengakses AWS sumber daya Anda. Contoh Corp juga memantau banyak lainnya Akun AWS untuk pelanggan lain. Anda dapat menggunakan IAM peran untuk membangun hubungan tepercaya antara akun Anda Akun AWS dan akun Example Corp. Salah satu aspek penting dari skenario ini adalah ID eksternal, informasi opsional yang dapat Anda gunakan dalam kebijakan kepercayaan IAM peran untuk menentukan

siapa yang dapat mengambil peran tersebut. Fungsi utama dari ID eksternal adalah mengatasi dan mencegah masalah deputi yang membingungkan.

Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggilan dapat dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses.

Pencegahan wakil kebingungan lintas akun

Diagram berikut menggambarkan masalah wakil kebingungan lintas akun.



Skenario ini mengasumsikan hal berikut:

- AWS 1 adalah Anda Akun AWS.
- AWS 1: ExampleRole adalah peran dalam akun Anda. Kebijakan kepercayaan peran ini mempercayai Example Corp dengan menentukan akun AWS Example Corp sebagai akun yang dapat mengambil peran tersebut.

Inilah yang terjadi:

1. Ketika Anda mulai menggunakan layanan Example Corp, Anda memberikan ARN dari AWS 1: ExampleRole ke Example Corp.
2. Contoh Corp menggunakan peran itu ARN untuk mendapatkan kredensi keamanan sementara untuk mengakses sumber daya di Anda. Akun AWS Dengan cara ini, Anda mempercayai Example Corp sebagai “deputi” yang dapat bertindak atas nama Anda.
3. AWS Pelanggan lain juga mulai menggunakan layanan Example Corp, dan pelanggan ini juga menyediakan ARN dari AWS 1: ExampleRole untuk Example Corp untuk digunakan. Agaknya pelanggan lain belajar atau menebak AWS 1: ExampleRole, yang bukan rahasia.

4. Ketika pelanggan lain meminta Example Corp untuk mengakses AWS sumber daya di (apa yang diklaimnya) akunnya, Example Corp menggunakan AWS 1: ExampleRole untuk mengakses sumber daya di akun Anda.

Inilah cara pelanggan lain bisa mendapatkan akses tanpa izin ke sumber daya Anda. Karena pelanggan lain mampu mengelabui Example Corp untuk melakukan tindakan di luar kesadaran pada sumber daya Anda, Example Corp kini adalah “confused deputy.”

Contoh Corp dapat mengatasi masalah wakil yang membingungkan dengan mengharuskan Anda memasukkan pemeriksaan `ExternalId` kondisi dalam kebijakan kepercayaan peran. Contoh Corp menghasilkan `ExternalId` nilai unik untuk setiap pelanggan dan menggunakan nilai itu dalam permintaannya untuk mengambil peran tersebut. `ExternalId`Nilai harus unik di antara pelanggan Example Corp dan dikendalikan oleh Example Corp, bukan pelanggannya. Inilah mengapa Anda mendapatkannya dari Example Corp dan Anda tidak membuatnya sendiri. Ini mencegah Example Corp menjadi wakil yang bingung dan memberikan akses ke sumber daya akun lain. AWS

Dalam skenario kami, bayangkan pengenal unik Example Corp untuk Anda adalah 12345, dan pengenalnya untuk pelanggan lain adalah 67890. Pengidentifikasi ini disederhanakan untuk skenario ini. Umumnya, pengidentifikasi ini adalahGUIDs. Dengan asumsi pengidentifikasi ini bersifat unik di antara pelanggan Example Corp, mereka adalah nilai yang masuk akal untuk digunakan bagi ID eksternal.

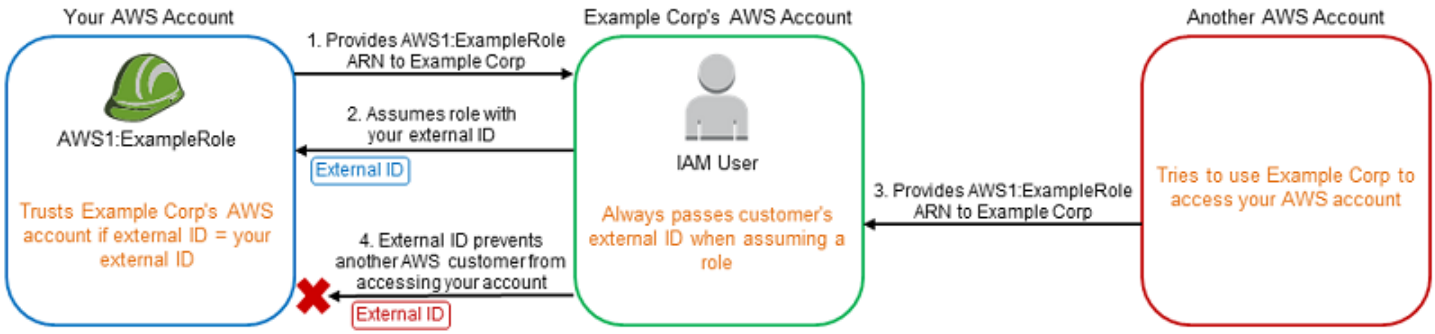
Contoh Corp memberikan nilai ID eksternal 12345 kepada Anda. Anda kemudian harus menambahkan elemen `Condition` ke kebijakan kepercayaan peran yang memerlukan nilai `sts:ExternalId` 12345, seperti ini:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
    }
  }
```

```
}
```

Elemen Kondisi dalam kebijakan ini memungkinkan Example Corp untuk mengambil peran hanya jika AssumeRole API panggilan menyertakan nilai ID eksternal 12345. Contoh Corp memastikan bahwa setiap kali mengambil peran atas nama pelanggan, selalu menyertakan nilai ID eksternal pelanggan dalam panggilan. AssumeRole Bahkan jika pelanggan lain memasok Example Corp dengan AndaARN, ia tidak dapat mengontrol ID eksternal yang disertakan Example Corp dalam permintaannya. AWS Hal ini membantu mencegah pelanggan yang tidak berwenang untuk mendapatkan akses ke sumber daya Anda.

Diagram berikut menggambarkannya.



1. Seperti sebelumnya, ketika Anda mulai menggunakan layanan Example Corp, Anda memberikan ARN dari AWS 1: ExampleRole to Example Corp.
2. Saat Example Corp menggunakan peran itu ARN untuk mengambil peran AWS 1: ExampleRole, Example Corp menyertakan ID eksternal Anda (12345) dalam panggilan. AssumeRole API ID eksternal cocok dengan kebijakan kepercayaan peran, sehingga AssumeRole API panggilan berhasil dan Example Corp memperoleh kredensi keamanan sementara untuk mengakses sumber daya di Anda. Akun AWS
3. AWS Pelanggan lain juga mulai menggunakan layanan Example Corp, dan seperti sebelumnya, pelanggan ini juga menyediakan ARN dari AWS 1: ExampleRole untuk Contoh Corp untuk digunakan.
4. Tapi kali ini, ketika Example Corp mencoba untuk mengambil peran AWS 1: ExampleRole, ia memberikan ID eksternal yang terkait dengan pelanggan lain (67890). Pelanggan lain tidak bisa mengubah ini. Contoh Corp melakukan ini karena permintaan untuk menggunakan peran berasal dari pelanggan lain, jadi 67890 menunjukkan keadaan di mana Example Corp bertindak. Karena Anda menambahkan kondisi dengan ID eksternal Anda sendiri (12345) ke kebijakan kepercayaan AWS 1: ExampleRole, AssumeRole API panggilan gagal. Pelanggan lain tidak bisa mendapatkan akses tanpa izin ke sumber daya dalam akun Anda (ditunjukkan oleh “X” merah di diagram).

ID eksternal membantu mencegah pelanggan lain menipu Example Corp agar tanpa disadari mengakses sumber daya Anda.

Pencegahan confused deputy lintas layanan

Sebaiknya gunakan kunci konteks kondisi `aws:SourceArn`aws:SourceAccountaws:SourceOrgID,, atau `aws:SourceOrgPaths`global dalam kebijakan berbasis sumber daya untuk membatasi izin yang dimiliki layanan ke sumber daya tertentu. Gunakan `aws:SourceArn` untuk mengaitkan hanya satu sumber daya dengan akses lintas layanan. Gunakan `aws:SourceAccount` untuk membiarkan sumber daya apa pun di akun itu dikaitkan dengan penggunaan lintas layanan. Gunakan `aws:SourceOrgID` untuk memungkinkan sumber daya apa pun dari akun apa pun dalam suatu organisasi dikaitkan dengan penggunaan lintas layanan. Gunakan `aws:SourceOrgPaths` untuk mengaitkan sumber daya apa pun dari akun dalam AWS Organizations jalur dengan penggunaan lintas layanan. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat Memahami jalur AWS Organizations entitas.

Cara paling terperinci untuk melindungi dari masalah wakil yang membingungkan adalah dengan menggunakan kunci konteks kondisi `aws:SourceArn` global dengan penuh ARN sumber daya dalam kebijakan berbasis sumber daya Anda. Jika Anda tidak mengetahui sumber daya yang lengkap ARN atau jika Anda menentukan beberapa sumber daya, gunakan kunci konteks kondisi `aws:SourceArn` global dengan wildcard (`*`) untuk bagian yang tidak diketahui dari file. ARN Misalnya, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Jika `aws:SourceArn` nilainya tidak berisi ID akun, seperti bucket Amazon S3ARN, Anda harus menggunakan keduanya `aws:SourceAccount` dan `aws:SourceArn` untuk membatasi izin.

Untuk melindungi dari masalah wakil yang membingungkan dalam skala besar, gunakan `aws:SourceOrgID` atau kunci konteks kondisi `aws:SourceOrgPaths` global dengan ID organisasi atau jalur organisasi sumber daya dalam kebijakan berbasis sumber daya Anda. Kebijakan yang menyertakan `aws:SourceOrgID` atau `aws:SourceOrgPaths` kunci akan secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbarui kebijakan secara manual saat menambahkan, menghapus, atau memindahkan akun di organisasi Anda.

Untuk kebijakan kepercayaan non-service-linked peran, setiap layanan dalam kebijakan kepercayaan telah melakukan `iam:PassRole` tindakan untuk memverifikasi bahwa peran tersebut berada di akun yang sama dengan layanan panggilan. Akibatnya, menggunakan `aws:SourceAccountaws:SourceOrgID`, atau `aws:SourceOrgPaths` dengan kebijakan kepercayaan tersebut tidak diperlukan. Menggunakan `aws:SourceArn` dalam kebijakan

kepercayaan memungkinkan Anda menentukan sumber daya peran yang dapat diasumsikan atas nama, seperti fungsi Lambda. ARN Beberapa kebijakan Layanan AWS penggunaan `aws:SourceAccount` dan `aws:SourceArn` kepercayaan untuk peran yang baru dibuat, tetapi menggunakan kunci tidak diperlukan untuk peran yang ada di akun Anda.

> **Note**
>
> Layanan AWS yang terintegrasi dengan AWS Key Management Service menggunakan hibah KMS kunci tidak mendukung kunci`aws:SourceArn`,`aws:SourceAccount`,`aws:SourceOrgID`, atau `aws:SourceOrgPaths` kondisi. Penggunaan kunci kondisi ini dalam kebijakan KMS kunci akan menghasilkan perilaku yang tidak terduga jika kunci juga digunakan oleh Layanan AWS via hibah KMS kunci.

Pencegahan deputi kebingungan lintas layanan untuk AWS Security Token Service

Banyak AWS layanan mengharuskan Anda menggunakan peran untuk memungkinkan layanan mengakses sumber daya layanan lain atas nama Anda. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut peran layanan. Peran memerlukan dua kebijakan: kebijakan kepercayaan peran yang menentukan prinsipal yang diizinkan untuk mengambil peran dan kebijakan izin yang menentukan apa yang dapat dilakukan dengan peran tersebut. Kebijakan kepercayaan peran adalah satu-satunya jenis kebijakan berbasis sumber daya di. IAM Lainnya Layanan AWS memiliki kebijakan berbasis sumber daya, seperti kebijakan bucket Amazon S3.

Ketika layanan mengambil peran atas nama Anda, kepala layanan harus diizinkan untuk melakukan `sts:AssumeRole`tindakan dalam kebijakan kepercayaan peran. Saat layanan memanggil`sts:AssumeRole`, AWS STS mengembalikan sekumpulan kredensional keamanan sementara yang digunakan prinsipal layanan untuk mengakses sumber daya yang diizinkan oleh kebijakan izin peran. Saat layanan mengambil peran di akun Anda, Anda dapat menyertakan kunci konteks kondisi `aws:SourceArn`aws:SourceAccount`,`aws:SourceOrgID`, atau `aws:SourceOrgPaths` global dalam kebijakan kepercayaan peran Anda untuk membatasi akses ke peran hanya untuk permintaan yang dihasilkan oleh sumber daya yang diharapkan.

Misalnya, di AWS Systems Manager Incident Manager, Anda harus memilih peran untuk mengizinkan Manajer Insiden menjalankan dokumen otomatisasi Systems Manager atas nama Anda. Dokumen otomatisasi dapat mencakup rencana respons otomatis untuk insiden yang diprakarsai oleh CloudWatch alarm atau peristiwa. EventBridge Dalam contoh kebijakan kepercayaan peran berikut,

Anda dapat menggunakan kunci `aws:SourceArn` kondisi untuk membatasi akses ke peran layanan berdasarkan catatan insiden. ARN Hanya catatan insiden yang dibuat dari sumber daya rencana respons `myresponseplan` yang dapat menggunakan peran ini.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "ssm-incidents.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ssm-incidents:*:111122223333:incident-
record/myresponseplan/*"
      }
    }
  }
}
```

> **Note**
>
> Tidak semua integrasi layanan dengan AWS STS dukunganaws:SourceArn,, aws:SourceAccountaws:SourceOrgID, atau kunci aws:SourceOrgPaths kondisi. Penggunaan kunci ini dalam kebijakan IAM kepercayaan dengan integrasi yang tidak didukung dapat mengakibatkan perilaku yang tidak terduga.

Skenario umum untuk IAM peran

Seperti kebanyakan AWS fitur, Anda biasanya memiliki dua cara untuk menggunakan peran: interaktif di IAM konsol, atau secara terprogram dengan AWS CLI, Alat untuk Windows PowerShell, atau. API

- IAMpengguna di akun Anda menggunakan IAM konsol dapat beralih ke peran untuk sementara menggunakan izin peran di konsol. Pengguna menyerahkan izin mereka dan mengambil izin yang ditetapkan untuk peran tersebut. Saat pengguna keluar dari peran, izin asli mereka dipulihkan.
- Aplikasi atau layanan yang ditawarkan oleh AWS (seperti AmazonEC2) dapat mengambil peran dengan meminta kredensil keamanan sementara untuk peran yang dapat digunakan untuk

membuat permintaan terprogram. AWS Anda menggunakan peran dengan cara ini sehingga Anda tidak perlu berbagi atau mempertahankan kredensil keamanan jangka panjang (misalnya, dengan membuat IAM pengguna) untuk setiap entitas yang memerlukan akses ke sumber daya.

> **Note**
>
> Panduan ini menggunakan frasa beralih ke peran dan memegang peran secara bergantian.

Cara termudah untuk menggunakan peran adalah dengan memberikan izin kepada IAM pengguna Anda untuk beralih ke peran yang Anda buat dalam peran Anda sendiri atau yang lain Akun AWS. Mereka dapat beralih peran dengan mudah menggunakan IAM konsol untuk menggunakan izin yang biasanya tidak Anda inginkan, dan kemudian keluar dari peran untuk menyerahkan izin tersebut. Ini dapat membantu mencegah akses yang tidak disengaja atau modifikasi sumber daya sensitif.

Untuk penggunaan peran yang lebih kompleks, seperti memberikan akses ke aplikasi dan layanan, atau pengguna eksternal gabungan, Anda dapat menghubungi file. `AssumeRole` API APIPanggilan ini mengembalikan satu set kredensi sementara yang dapat digunakan aplikasi dalam panggilan berikutnyaAPI. Tindakan yang dicoba dengan kredensial sementara hanya memiliki izin yang diberikan oleh peran terkait. Sebuah aplikasi tidak harus "keluar" dari peran sebagaimana pengguna di konsol; melainkan aplikasi hanya berhenti menggunakan kredensial sementara dan melanjutkan melakukan panggilan dengan kredensial yang asli.

Pengguna gabungan masuk dengan menggunakan kredensil dari penyedia identitas (iDP). AWS kemudian memberikan kredensi sementara ke iDP tepercaya untuk diteruskan ke pengguna untuk disertakan dalam permintaan sumber daya berikutnya. AWS Kredensial tersebut memberikan izin yang diberikan kepada peran yang ditetapkan.

Bagian ini memberikan gambaran tentang skenario berikut:

- Menyediakan akses untuk IAM pengguna di salah satu Akun AWS yang Anda miliki untuk mengakses sumber daya di akun lain yang Anda miliki
- Menyediakan akses ke non beban AWS kerja
- Menyediakan akses ke IAM pengguna yang Akun AWS dimiliki oleh pihak ketiga
- Menyediakan akses untuk layanan yang ditawarkan oleh AWS ke AWS sumber daya
- Menyediakan akses untuk pengguna yang diautentikasi secara eksternal (federasi identitas)

Akses untuk IAM pengguna lain Akun AWS yang Anda miliki

Anda dapat memberikan izin kepada IAM pengguna untuk beralih ke peran dalam peran Anda Akun AWS atau ke peran yang ditentukan di peran lain Akun AWS yang Anda miliki.

> **Note**
>
> Jika Anda ingin memberikan akses ke akun yang tidak Anda miliki atau kontrol, lihat Akses ke Akun AWS yang dimiliki oleh pihak ketiga nanti dalam topik ini.

Bayangkan Anda memiliki EC2 contoh Amazon yang penting bagi organisasi Anda. Alih-alih secara langsung memberikan izin kepada pengguna Anda untuk menghentikan instans, Anda dapat membuat peran atas hak istimewa tersebut. Kemudian, izinkan administrator untuk beralih ke peran saat mereka perlu menghentikan suatu instans. Melakukan hal ini akan menambah lapisan perlindungan berikut ke kejadian:

- Anda harus secara tegas memberikan izin kepada pengguna Anda untuk mengambil peran tersebut.
- Pengguna Anda harus secara aktif beralih ke peran menggunakan AWS Management Console atau mengambil peran menggunakan AWS CLI atau AWS API.
- Anda dapat menambahkan perlindungan multi-faktor authentication (MFA) ke peran sehingga hanya pengguna yang masuk dengan MFA perangkat yang dapat mengambil peran tersebut. Untuk mempelajari cara mengonfigurasi peran sehingga pengguna yang mengambil peran harus terlebih dahulu diautentikasi menggunakan otentikasi multi-faktor (MFA), lihat. APIAkses aman dengan MFA

Kami menyarankan menggunakan pendekatan ini untuk menegakkan prinsip hak istimewa paling rendah. Ini berarti membatasi penggunaan izin yang ditingkatkan hanya pada saat diperlukan untuk tugas tertentu. Dengan peran Anda dapat membantu mencegah perubahan yang tidak disengaja terhadap lingkungan yang sensitif, terutama jika Anda menggabungkannya dengan audit untuk membantu memastikan bahwa peran tersebut hanya digunakan saat diperlukan.

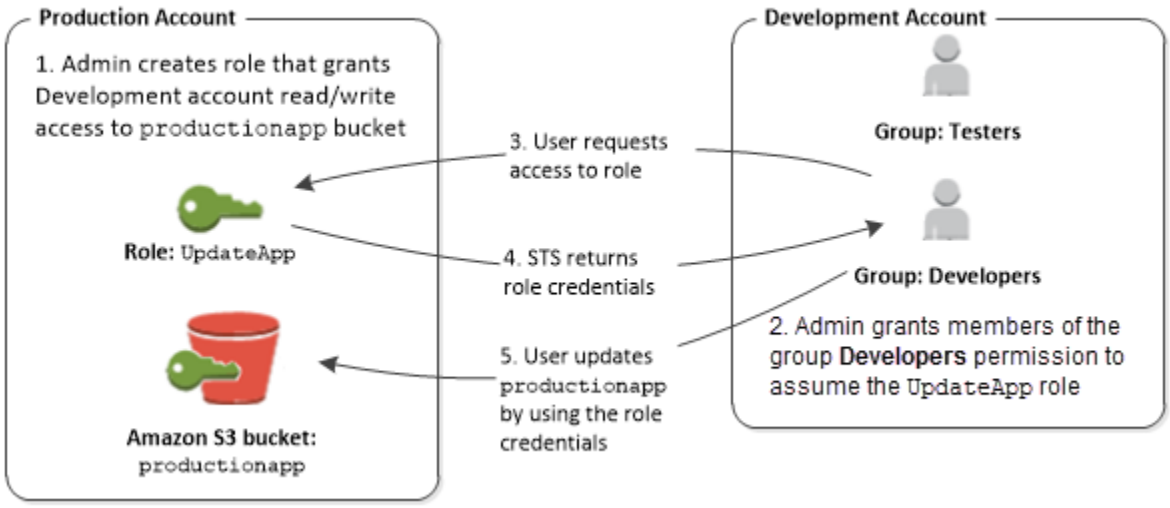
Saat Anda membuat peran untuk tujuan ini, Anda menetapkan akun berdasarkan ID yang penggunanya memerlukan akses dalam elemen `Principal` kebijakan kepercayaan peran. Kemudian, Anda dapat memberikan izin kepada pengguna tertentu dalam akun lain untuk beralih ke peran tersebut. Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda

(organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? .

Pengguna dalam satu akun dapat beralih ke peran dalam akun yang sama atau berbeda. Saat menggunakan peran tersebut, pengguna hanya dapat melakukan tindakan dan mengakses sumber daya yang diizinkan dengan peran tersebut; izin pengguna yang asli ditangguhkan. Saat pengguna keluar dari peran, izin pengguna asli mereka dipulihkan.

Contoh skenario menggunakan akun pengembangan dan produksi terpisah

Bayangkan bahwa organisasi Anda memiliki banyak Akun AWS untuk mengisolasi lingkungan pengembangan dari lingkungan produksi. Pengguna di akun pengembangan mungkin kadang kala perlu mengakses sumber daya di akun produksi. Misalnya, Anda mungkin memerlukan akses akun silang saat mempromosikan pembaruan dari lingkungan pengembangan ke lingkungan produksi. Meskipun Anda dapat membuat identitas terpisah (dan kata sandi) untuk pengguna yang bekerja di kedua akun, mengelola kredensial untuk beberapa akun membuat manajemen identitas menjadi sulit. Dalam gambar berikut, semua pengguna dikelola dalam akun pengembangan, tetapi beberapa pengembang memerlukan akses terbatas ke akun produksi. Akun pengembangan memiliki dua grup: Penguji dan Developer, lalu setiap grup memiliki kebijakannya sendiri.



1. Di akun produksi, administrator menggunakan IAM untuk membuat `UpdateApp` peran di akun itu. Dalam peran tersebut, administrator mendefinisikan kebijakan kepercayaan yang menentukan akun pengembangan sebagai `Principal`, yang berarti bahwa pengguna sah dari akun pengembangan dapat menggunakan peran `UpdateApp`. Administrator juga menetapkan kebijakan izin untuk peran yang menentukan izin baca dan tulis ke bucket Amazon S3 bernama. `productionapp`

Kemudian, administrator membagikan informasi yang sesuai dengan siapa pun yang perlu menjalankan peran tersebut. Informasi tersebut adalah nomor akun dan nama peran (untuk pengguna AWS konsol) atau Nama Sumber Daya Amazon (ARN) (untuk AWS CLI atau AWS API akses). Peran ARN mungkin terlihat seperti`arn:aws:iam::123456789012:role/UpdateApp`, di mana peran diberi nama `UpdateApp` dan peran dibuat di nomor akun 123456789012.

> **Note**
>
> Administrator dapat mengonfigurasi peran secara opsional sehingga pengguna yang mengambil peran harus terlebih dahulu diautentikasi menggunakan otentikasi multi-faktor (). MFA Untuk informasi selengkapnya, lihat APIAkses aman dengan MFA.

2. Dalam akun pengembangan, administrator memberi anggota grup Pengembang izin untuk beralih ke peran. Ini dilakukan dengan memberikan izin kepada grup Pengembang untuk memanggil AWS Security Token Service (AWS STS) `AssumeRole` API untuk `UpdateApp` peran tersebut. Setiap IAM pengguna yang termasuk dalam grup Pengembang di akun pengembangan sekarang dapat beralih ke `UpdateApp` peran di akun produksi. Pengguna lain yang tidak berada dalam grup developer tidak memiliki izin untuk beralih ke peran tersebut sehingga tidak dapat mengakses bucket S3 dalam akun produksi.
3. Pengguna meminta untuk beralih ke peran:
   - AWS konsol: Pengguna memilih nama akun di bilah navigasi dan memilih Beralih Peran. Pengguna menentukan ID akun (atau alias) dan nama peran. Atau, pengguna dapat mengklik tautan yang dikirim melalui email oleh administrator. Tautan tersebut membawa pengguna ke halaman Beralih Peran dengan detail yang sudah diisi.
   - AWS API/AWS CLI: Seorang pengguna dalam grup Pengembang dari akun pengembangan memanggil `AssumeRole` fungsi untuk mendapatkan kredensi untuk peran tersebut`UpdateApp`. Pengguna menentukan `UpdateApp` peran ARN sebagai bagian dari panggilan. Jika pengguna dalam grup Penguji membuat permintaan yang sama, permintaan gagal karena Penguji tidak memiliki izin `AssumeRole` untuk memanggil peran tersebut`UpdateApp`. ARN
4. AWS STS mengembalikan kredensi sementara:
   - AWS console: AWS STS memverifikasi permintaan dengan kebijakan kepercayaan peran untuk memastikan bahwa permintaan tersebut berasal dari entitas tepercaya (yaitu: akun pengembangan). Setelah verifikasi, AWS STS mengembalikan kredensi keamanan sementara ke konsol. AWS

- API/CLI: AWS STS memverifikasi permintaan terhadap kebijakan kepercayaan peran untuk memastikan bahwa permintaan tersebut berasal dari entitas tepercaya (yaitu: akun Pengembangan). Setelah verifikasi, AWS STS mengembalikan kredensi keamanan sementara ke aplikasi.

5. Kredensi sementara memungkinkan akses ke sumber daya: AWS

- AWS konsol: AWS Konsol menggunakan kredensi sementara atas nama pengguna untuk semua tindakan konsol berikutnya, dalam hal ini, untuk membaca dan menulis ke bucket. `productionapp` Konsol tidak dapat mengakses sumber daya lain di akun produksi. Saat pengguna keluar dari peran, izin pengguna kembali ke izin awal yang ada sebelum beralih ke peran.
- API/CLI: Aplikasi ini menggunakan kredensi keamanan sementara untuk memperbarui bucket. `productionapp` Dengan kredensial keamanan sementara, aplikasi hanya dapat membaca dari dan menulis ke bucket `productionapp` dan tidak dapat mengakses sumber daya lain dalam akun Produksi. Aplikasi tidak harus keluar dari peran, tetapi berhenti menggunakan kredenal sementara dan menggunakan kredensi asli dalam panggilan berikutnya. API

Sumber daya tambahan

Untuk informasi selengkapnya, lihat informasi berikut:

- IAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran

Akses untuk non beban AWS kerja

IAMPeran adalah objek di AWS Identity and Access Management (IAM) yang diberi izin. Ketika Anda mengasumsikan peran tersebut menggunakan IAM identitas atau identitas dari luar AWS, itu memberi Anda kredensi keamanan sementara untuk sesi peran Anda. Anda mungkin memiliki beban kerja yang berjalan di pusat data atau infrastruktur lain di luar AWS yang harus mengakses AWS sumber daya Anda. Alih-alih membuat, mendistribusikan, dan mengelola kunci akses jangka panjang, Anda dapat menggunakan AWS Identity and Access Management Peran Di Mana Saja (IAMPeran Di Mana Saja) untuk mengautentikasi non AWS beban kerja Anda. IAMRoles Anywhere menggunakan sertifikat X.509 dari otoritas sertifikat (CA) Anda untuk mengautentikasi identitas dan menyediakan akses dengan aman Layanan AWS dengan kredenal sementara yang disediakan oleh peran. IAM

Untuk menggunakan IAM Peran Di Mana Saja

1. Siapkan CA menggunakan AWS Private Certificate Authorityatau menggunakan CA dari PKI infrastruktur Anda sendiri.
2. Setelah menyiapkan CA, Anda membuat objek di IAM Roles Anywhere yang disebut jangkar kepercayaan. Anchor ini membangun kepercayaan antara IAM Roles Anywhere dan CA Anda untuk otentikasi.
3. Anda kemudian dapat mengonfigurasi IAM peran yang ada, atau membuat peran baru yang mempercayai layanan IAM Roles Anywhere.
4. Otentikasi non AWS beban kerja Anda dengan IAM Roles Anywhere menggunakan jangkar kepercayaan. AWS memberikan kredensil sementara non AWS beban kerja untuk IAM peran yang memiliki akses sumber daya Anda. AWS

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang menyediakan akses ke AWS non-beban kerja.

- Untuk informasi selengkapnya tentang mengonfigurasi IAM Peran Di Mana Saja, lihat Apa itu AWS Identity and Access Management Peran Di Mana Saja dalam Panduan Pengguna IAM Peran Di Mana Saja.
- Untuk mempelajari cara menyiapkan infrastruktur kunci publik (PKI) untuk IAM Peran Di Mana Saja, lihat IAMPeran Di Mana Saja dengan otoritas sertifikat eksternal di Blog AWS Keamanan.

Akses ke Akun AWS yang dimiliki oleh pihak ketiga

Ketika pihak ketiga memerlukan akses ke AWS sumber daya organisasi Anda, Anda dapat menggunakan peran untuk mendelegasikan akses ke sumber daya organisasi Anda. Misalnya, pihak ketiga mungkin menyediakan layanan untuk mengelola sumber daya AWS Anda. Dengan IAM peran, Anda dapat memberikan pihak ketiga ini akses ke AWS sumber daya Anda tanpa membagikan kredensi AWS keamanan Anda. Sebaliknya, pihak ketiga dapat mengakses AWS sumber daya Anda dengan mengasumsikan peran yang Anda buat di situs Anda Akun AWS. Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? .

Pihak ketiga harus menyediakan Anda dengan informasi berikut untuk membuat peran yang dapat mereka tanggung:

- Akun AWS ID pihak ketiga. Anda menentukan Akun AWS ID mereka sebagai prinsipal saat Anda menentukan kebijakan kepercayaan untuk peran tersebut.
- ID eksternal untuk mengasosiasikan secara unik dengan peran. ID eksternal dapat berupa pengenal apa pun yang hanya diketahui oleh Anda dan pihak ketiga. Misalnya, Anda dapat menggunakan ID faktur antara Anda dan pihak ketiga, tetapi tidak menggunakan sesuatu yang dapat ditebak, seperti nama atau nomor telepon pihak ketiga. Anda harus menentukan ID ini saat Anda menentukan kebijakan kepercayaan untuk peran tersebut. Pihak ketiga harus memberikan ID ini saat memegang peran tersebut.
- Izin yang diperlukan pihak ketiga untuk bekerja dengan AWS sumber daya Anda. Anda harus menentukan izin ini saat menentukan kebijakan izin peran. Kebijakan ini menetapkan tindakan apa yang dapat mereka lakukan dan sumber daya apa yang dapat mereka akses.

Setelah membuat peran, Anda harus memberikan Amazon Resource Name (ARN) peran tersebut kepada pihak ketiga. Mereka membutuhkan peran Anda untuk mengambil peran. ARN

> **Important**
>
> Saat Anda memberi pihak ketiga akses ke AWS sumber daya Anda, mereka dapat mengakses sumber daya apa pun yang Anda tentukan dalam kebijakan. Penggunaan sumber daya Anda oleh mereka dibebankan kepada Anda. Pastikan bahwa Anda membatasi penggunaan sumber daya Anda dengan tepat.

Eksternal IDs untuk akses pihak ketiga

ID eksternal memungkinkan pengguna yang mengambil peran untuk menegaskan keadaan di mana mereka beroperasi. Itu ini juga memberikan cara bagi pemilik akun untuk mengizinkan peran tersebut untuk diasumsikan hanya dalam keadaan tertentu. Fungsi utama dari ID eksternal adalah untuk mengatasi dan mencegahMasalah confused deputy.

> **Important**
>
> AWS tidak memperlakukan ID eksternal sebagai rahasia. Setelah Anda membuat rahasia seperti access key pair atau password di AWS, Anda tidak dapat melihatnya lagi. ID eksternal untuk peran dapat dilihat oleh siapa pun yang memiliki izin untuk melihat peran tersebut.

Kapan saya harus menggunakan ID eksternal?

Gunakan ID eksternal dalam situasi-situasi berikut:

- Anda adalah Akun AWS pemilik dan Anda telah mengonfigurasi peran untuk pihak ketiga yang mengakses orang lain Akun AWS selain milik Anda. Anda harus meminta ID eksternal dari pihak ketiga yang disertakannya, ketika mereka mengasumsikan peran Anda. Kemudian Anda memeriksa ID eksternal tersebut dalam kebijakan kepercayaan peran Anda. Melakukannya memastikan bahwa pihak eksternal dapat mengasumsikan peran Anda hanya ketika bertindak atas nama Anda.
- Anda berada dalam posisi mengasumsikan peran atas nama pelanggan yang berbeda seperti Example Corp dalam skenario kami sebelumnya. Anda harus menetapkan ID eksternal unik untuk setiap pelanggan dan menginstruksikan mereka untuk menambahkan ID eksternal ke kebijakan kepercayaan peran mereka. Kemudian Anda harus memastikan bahwa Anda selalu menyertakan ID eksternal yang benar dalam permintaan Anda untuk mengasumsikan peran.

  Anda mungkin sudah memiliki pengidentifikasi unik untuk setiap pelanggan Anda, dan ID unik ini cukup untuk digunakan sebagai ID eksternal. ID eksternal bukan merupakan nilai khusus yang Anda perlukan untuk membuat secara eksplisit, atau melacak secara terpisah, hanya untuk tujuan ini.

  Anda harus selalu menentukan ID eksternal dalam `AssumeRole` API panggilan Anda. Selain itu ketika pelanggan memberi Anda peranARN, uji apakah Anda dapat mengambil peran baik dengan dan tanpa ID eksternal yang benar. Jika Anda dapat mengambil peran tanpa ID eksternal yang benar, jangan menyimpan peran pelanggan ARN dalam sistem Anda. Tunggu hingga pelanggan Anda memperbarui kebijakan kepercayaan peran untuk meminta ID eksternal yang benar. Dengan cara ini, Anda membantu pelanggan Anda untuk melakukan hal yang benar, yang membantu Anda semua melindungi diri dari masalah confused deputy.

Contoh skenario menggunakan ID eksternal

Misalnya, katakanlah Anda memutuskan untuk menyewa perusahaan pihak ketiga bernama Example Corp untuk memantau Anda Akun AWS dan membantu mengoptimalkan biaya. Untuk melacak pengeluaran harian Anda, Example Corp perlu mengakses AWS sumber daya Anda. Example Corp juga memantau banyak akun AWS lain untuk nasabah lain.

Jangan berikan Example Corp akses ke IAM pengguna dan kredensialnya jangka panjang di akun Anda. AWS Sebagai gantinya, gunakan IAM peran dan kredensi keamanan sementaranya. IAMPeran

menyediakan mekanisme untuk memungkinkan pihak ketiga mengakses AWS sumber daya Anda tanpa perlu berbagi kredensi jangka panjang (seperti kunci akses IAM pengguna).

Anda dapat menggunakan IAM peran untuk membangun hubungan tepercaya antara akun Anda Akun AWS dan akun Example Corp. Setelah hubungan ini terjalin, anggota akun Example Corp dapat menghubungi AWS Security Token Service AssumeRoleAPIuntuk mendapatkan kredensi keamanan sementara. Anggota Example Corp kemudian dapat menggunakan kredensialnya untuk mengakses AWS sumber daya di akun Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang operasi AssumeRole dan AWS API operasi lain yang dapat Anda hubungi untuk mendapatkan kredensi keamanan sementara, lihat. Bandingkan AWS STS kredensialnya

Berikut rincian skenario ini secara lebih detail:

1. Anda merekrut Example Corp, sehingga mereka membuat pengenal pelanggan unik untuk Anda. Mereka memberi Anda ID pelanggan unik ini dan Akun AWS nomor mereka. Anda memerlukan informasi ini untuk membuat IAM peran di langkah berikutnya.

   > **Note**
   >
   > Contoh Corp dapat menggunakan nilai string apa pun yang mereka inginkan untuk ExternalId, asalkan unik untuk setiap pelanggan. Itu dapat berupa nomor akun pelanggan atau bahkan string karakter acak, selama tidak ada dua pelanggan yang memiliki nilai yang sama. Itu tidak dimaksudkan untuk menjadi 'rahasia'. Contoh Corp harus memberikan ExternalId nilai kepada setiap pelanggan. Yang penting adalah bahwa itu harus dihasilkan oleh Example Corp dan bukan pelanggan mereka untuk memastikan setiap ID eksternal unik.

2. Anda masuk AWS dan membuat IAM peran yang memberi Example Corp akses ke sumber daya Anda. Seperti IAM peran apa pun, peran tersebut memiliki dua kebijakan, kebijakan izin dan kebijakan kepercayaan. Kebijakan kepercayaan peran menentukan siapa yang dapat mengasumsikan peran tersebut. Dalam skenario sampel kami, kebijakan menentukan Akun AWS jumlah Example Corp sebagai. `Principal` Hal ini memungkinkan identitas dari akun tersebut untuk mengambil peran. Selain itu, Anda menambahkan elemen `Condition` pada kebijakan

kepercayaan. Ini `Condition` menguji `ExternalId` kunci konteks untuk memastikan bahwa itu sesuai dengan ID pelanggan unik dari Example Corp. Misalnya:

```
   "Principal": {"AWS": "Example Corp's Akun AWS ID"},
   "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. Kebijakan izin untuk peran menentukan apa saja yang diperbolehkan oleh peran untuk dilakukan seseorang. Misalnya, Anda dapat menentukan bahwa peran tersebut memungkinkan seseorang untuk mengelola hanya RDS sumber daya Amazon EC2 dan Amazon Anda tetapi bukan IAM pengguna atau grup Anda. Dalam skenario sampel kami, Anda menggunakan kebijakan izin untuk memberikan akses hanya-baca kepada Example Corp ke semua sumber daya dalam akun Anda.
4. Setelah membuat peran, Anda memberikan Amazon Resource Name (ARN) peran tersebut ke Example Corp.
5. Ketika Example Corp perlu mengakses AWS sumber daya Anda, seseorang dari perusahaan menelepon. AWS `sts:AssumeRole` API Panggilan tersebut ARN mencakup peran yang akan diasumsikan dan ExternalId parameter yang sesuai dengan ID pelanggan mereka.

Jika permintaan berasal dari seseorang yang menggunakan Example Corp Akun AWS, dan jika peran ARN dan ID eksternal sudah benar, permintaan berhasil. Ini kemudian memberikan kredensi keamanan sementara yang dapat digunakan Example Corp untuk mengakses AWS sumber daya yang diizinkan oleh peran Anda.

Dengan kata lain, ketika kebijakan peran mencakup ID eksternal, siapa pun yang ingin mengasumsikan peran tersebut haruslah prinsipal dalam peran tersebut dan harus menyertakan ID eksternal yang benar.

Poin kunci untuk eksternal IDs

- Dalam lingkungan multi-tenant di mana Anda mendukung beberapa pelanggan dengan AWS akun yang berbeda, kami sarankan untuk menggunakan satu ID eksternal per. Akun AWS ID ini harus berupa string acak yang dihasilkan oleh pihak ketiga.
- Untuk mewajibkan pihak ketiga untuk memberikan ID eksternal saat mengasumsikan peran, perbarui kebijakan kepercayaan peran tersebut dengan ID eksternal pilihan Anda.
- Untuk memberikan ID eksternal saat Anda mengambil peran, gunakan AWS CLI atau AWS API untuk mengambil peran tersebut. Untuk informasi selengkapnya, lihat STS AssumeRoleAPIoperasi, atau operasi peran STS asumsiCLI.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga.

- Untuk mempelajari cara mengizinkan orang lain melakukan tindakan dalam diri Anda Akun AWS, lihatMembuat peran menggunakan kebijakan kepercayaan khusus .
- Untuk mempelajari cara memberikan izin untuk beralih ke peran, lihat Berikan izin pengguna untuk beralih peran
- Untuk mempelajari cara membuat dan menyediakan kredensi keamanan sementara kepada pengguna tepercaya, Izin untuk kredensial keamanan sementara

Akses ke AWS layanan

Banyak AWS layanan mengharuskan Anda menggunakan peran untuk mengontrol apa yang dapat diakses oleh layanan tersebut. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut peran layanan. Ketika suatu peran melayani tujuan khusus untuk suatu layanan, itu dapat dikategorikan sebagai peran terkait layanan. Lihat dokumentasi AWS untuk setiap layanan guna melihat apakah layanan menggunakan peran dan untuk pelajari cara menetapkan peran untuk digunakan layanan.

Untuk detail tentang membuat peran untuk mendelegasikan akses ke layanan yang ditawarkan oleh AWS, lihatMembuat peran untuk mendelegasikan izin ke layanan AWS.

Akses ke pengguna yang diautentikasi secara eksternal (federasi identitas)

Pengguna Anda mungkin sudah memiliki identitas di luar AWS, seperti di direktori perusahaan Anda. Jika pengguna tersebut perlu bekerja dengan AWS sumber daya (atau bekerja dengan aplikasi yang mengakses sumber daya tersebut), maka pengguna tersebut juga memerlukan AWS kredensil keamanan. Anda dapat menggunakan IAM peran untuk menentukan izin bagi pengguna yang identitasnya digabungkan dari organisasi Anda atau penyedia identitas pihak ketiga (iDP).

> **Note**
>
> Sebagai praktik keamanan terbaik, kami sarankan Anda mengelola akses pengguna di Pusat IAM Identitas dengan federasi identitas alih-alih membuat IAM pengguna. Untuk informasi tentang situasi tertentu di mana IAM pengguna diperlukan, lihat Kapan membuat IAM pengguna (bukan peran).

Pengguna federasi dari aplikasi seluler atau berbasis web dengan Amazon Cognito

Jika Anda membuat aplikasi seluler atau berbasis web yang mengakses AWS sumber daya, aplikasi memerlukan kredensi keamanan untuk membuat permintaan terprogram. AWS Untuk sebagian besar skenario aplikasi seluler, kami menyarankan agar Anda menggunakan Amazon Cognito. Anda dapat menggunakan layanan ini dengan AWS Mobile SDK untuk iOS dan Mobile untuk Android dan Fire OS SDK untuk membuat identitas unik bagi pengguna dan mengautentikasi mereka untuk akses aman ke sumber daya Anda AWS .AWS Amazon Cognito mendukung penyedia identitas yang sama seperti yang tercantum di bagian berikutnya, dan juga mendukung identitas yang diautentikasi pengembang dan akses (tamu) yang tidak terotentikasi. Amazon Cognito juga menyediakan API operasi untuk menyinkronkan data pengguna sehingga dipertahankan saat pengguna berpindah antar perangkat. Untuk informasi selengkapnya, lihat Amazon Cognito untuk aplikasi seluler.

Menggabungkan pengguna dengan penyedia layanan identitas publik atau OpenID Connect

Bila memungkinkan, gunakan Amazon Cognito untuk skenario aplikasi berbasis web dan seluler. Amazon Cognito melakukan sebagian besar behind-the-scenes pekerjaan dengan layanan penyedia identitas publik untuk Anda. Ia bekerja dengan layanan pihak ketiga yang sama dan juga mendukung masuk secara anonim. Namun, untuk skenario yang lebih canggih, Anda dapat bekerja langsung dengan layanan pihak ketiga seperti Login with Amazon, Facebook, Google, atau IDP apa pun yang kompatibel dengan OpenID Connect (). OIDC Untuk informasi selengkapnya tentang penggunaan OIDC federasi menggunakan salah satu layanan ini, lihatOIDCfederasi.

Menggabungkan pengguna dengan 2.0 SAML

Jika organisasi Anda sudah menggunakan paket perangkat lunak penyedia identitas yang mendukung SAML 2.0 (Security Assertion Markup Language 2.0), Anda dapat membuat kepercayaan antara organisasi Anda sebagai penyedia identitas (iDP) dan AWS sebagai penyedia layanan. Anda kemudian dapat menggunakan SAML untuk menyediakan pengguna Anda dengan federasi single-sign on (SSO) ke AWS Management Console atau akses federasi ke operasi panggilan. AWS API Misalnya, jika perusahaan Anda menggunakan Microsoft Active Directory dan Active Directory Federation Services, maka Anda dapat melakukan federasi menggunakan SAML 2.0. Untuk informasi selengkapnya tentang federasi pengguna dengan SAML 2.0, lihatSAML2.0 federasi.

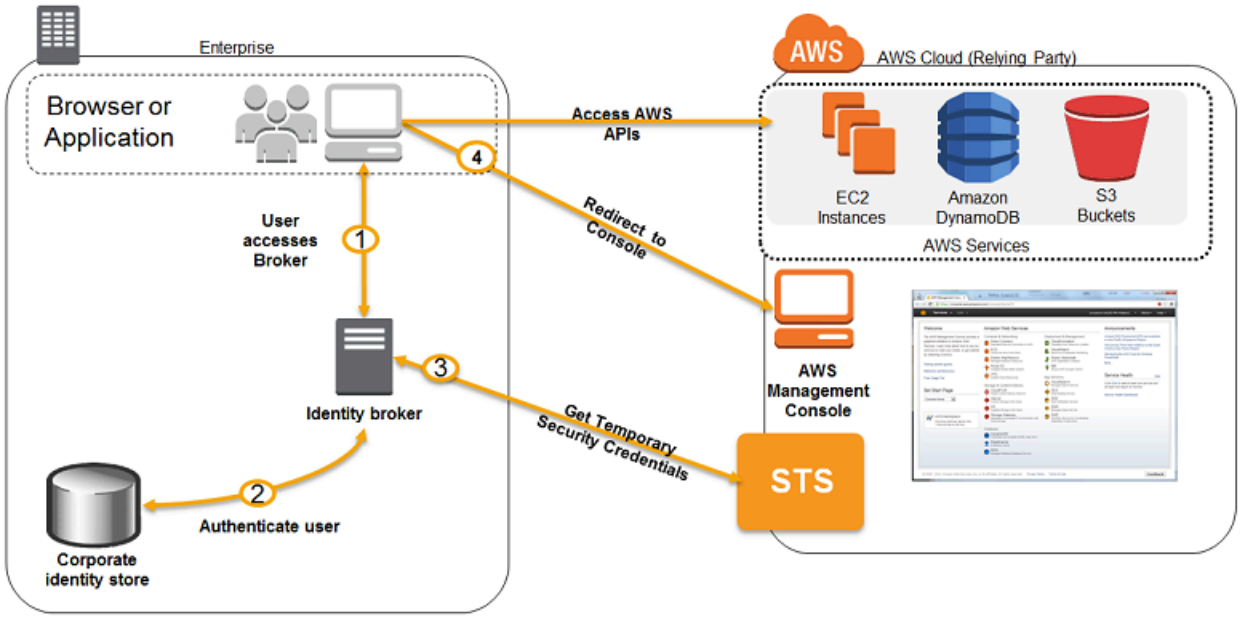
Pengguna federasi dengan membuat aplikasi broker identitas khusus

Jika toko identitas Anda tidak kompatibel dengan SAML 2.0, maka Anda dapat membangun aplikasi broker identitas khusus untuk melakukan fungsi serupa. Aplikasi broker mengotentikasi pengguna, meminta kredensi sementara untuk pengguna dari AWS, dan kemudian menyediakannya kepada pengguna untuk mengakses sumber daya. AWS

Misalnya, Example Corp. memiliki banyak karyawan yang perlu menjalankan aplikasi internal yang mengakses AWS sumber daya perusahaan. Karyawan sudah memiliki identitas dalam identitas perusahaan dan sistem otentikasi, dan Example Corp. tidak ingin membuat IAM pengguna terpisah untuk setiap karyawan perusahaan.

Bob adalah pengembang di Example Corp. Untuk mengaktifkan aplikasi internal Example Corp untuk mengakses AWS sumber daya perusahaan, Bob mengembangkan aplikasi broker identitas kustom. Aplikasi memverifikasi bahwa karyawan masuk ke sistem identitas dan otentikasi Example Corp. yang ada, yang mungkin menggunakan, Active DirectoryLDAP, atau sistem lain. Aplikasi broker identitas kemudian mendapatkan kredensial keamanan sementara bagi karyawan. Skenario ini mirip dengan yang sebelumnya (aplikasi seluler yang menggunakan sistem otentikasi khusus), kecuali bahwa aplikasi yang membutuhkan akses ke AWS sumber daya semuanya berjalan dalam jaringan perusahaan, dan perusahaan memiliki sistem otentikasi yang ada.

Untuk mendapatkan kredensial keamanan sementara, aplikasi broker identitas menelepon `AssumeRole` atau `GetFederationToken` untuk memperoleh kredensial keamanan sementara, tergantung pada bagaimana Bob ingin mengelola kebijakan untuk pengguna dan kapan kredensial sementara harus kedaluwarsa. (Untuk informasi lebih lanjut tentang perbedaan antara API operasi ini, lihat Kredensi keamanan sementara di IAM danIzin untuk kredensial keamanan sementara.) Panggilan mengembalikan kredensil keamanan sementara yang terdiri dari ID kunci AWS akses, kunci akses rahasia, dan token sesi. Aplikasi broker identitas membuat kredensial keamanan sementara ini tersedia bagi aplikasi internal perusahaan. Aplikasi kemudian dapat menggunakan kredensial sementara untuk membuat panggilan ke AWS secara langsung. Aplikasi menyimpan kredensial hingga kedaluwarsa, kemudian meminta set kredensial sementara yang baru. Gambar berikut mengilustrasikan skenario ini.



Skenario ini memiliki atribut berikut:

- Aplikasi broker identitas memiliki izin untuk mengakses IAM layanan token (STS) API untuk membuat kredensil keamanan sementara.
- Aplikasi broker identitas dapat memverifikasi bahwa karyawan diautentikasi dengan sistem autentikasi yang sudah ada.
- Pengguna bisa mendapatkan sementara URL yang memberi mereka akses ke Konsol AWS Manajemen (yang disebut sebagai sistem masuk tunggal).

Untuk informasi tentang membuat kredensial keamanan sementara, lihat Bandingkan AWS STS kredensialnya. Untuk informasi selengkapnya tentang pengguna federasi yang mendapatkan akses ke AWS Management Console, lihatMengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console.

IAMpenciptaan peran

Untuk membuat peran, Anda dapat menggunakan AWS CLI, Alat untuk Windows PowerShell, atau IAMAPI. AWS Management Console

Jika Anda menggunakan AWS Management Console, wizard memandu Anda melalui langkah-langkah untuk membuat peran. Wizard memiliki langkah yang sedikit berbeda tergantung pada apakah Anda membuat peran untuk AWS layanan, untuk Akun AWS, atau untuk pengguna federasi.

Peran untuk IAM pengguna

Buat peran ini untuk mendelegasikan izin dalam peran Anda Akun AWS atau ke peran yang ditentukan di peran lain Akun AWS yang Anda miliki. Pengguna dalam satu akun dapat beralih ke peran dalam akun yang sama atau berbeda. Saat menggunakan peran tersebut, pengguna hanya dapat melakukan tindakan dan mengakses sumber daya yang diizinkan dengan peran tersebut; izin pengguna yang asli ditangguhkan. Saat pengguna keluar dari peran, izin pengguna asli mereka dipulihkan.

Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke pengguna IAM.

Untuk informasi selengkapnya tentang membuat peran untuk akses lintas akun, lihatMembuat peran menggunakan kebijakan kepercayaan khusus .

Peran untuk AWS layanan

Buat peran ini untuk mendelegasikan izin ke layanan yang dapat melakukan tindakan atas nama Anda. Peran layanan yang Anda berikan ke layanan harus memiliki IAM kebijakan dengan izin yang memungkinkan layanan melakukan tindakan yang terkait dengan layanan tersebut. Izin yang berbeda diperlukan untuk setiap AWS layanan.

Untuk informasi selengkapnya tentang membuat peran layanan, lihatMembuat peran untuk mendelegasikan izin ke layanan AWS.

Untuk informasi selengkapnya tentang membuat peran terkait layanan, lihat. Buat peran tertaut layanan

Peran untuk federasi identitas

Buat peran ini untuk mendelegasikan izin kepada pengguna yang sudah memiliki identitas di luar. AWS Saat Anda menggunakan penyedia identitas , Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Pengguna eksternal Anda masuk melalui iDP, dan Anda dapat memberikan izin identitas eksternal tersebut untuk menggunakan AWS sumber daya di akun Anda. Penyedia identitas membantu menjaga keamanan AWS akun Anda karena Anda tidak perlu mendistribusikan atau menyematkan kredensi keamanan jangka panjang, seperti kunci akses, di aplikasi Anda.

Untuk informasi selengkapnya, lihat Buat peran untuk penyedia identitas pihak ketiga (federasi).

Membuat peran untuk mendelegasikan izin ke pengguna IAM

Anda dapat menggunakan IAM peran untuk mendelegasikan akses ke AWS sumber daya. Dengan IAM peran, Anda dapat membangun hubungan kepercayaan antara akun kepercayaan Anda dan lainnya AWS akun tepercaya. Akun trusting memiliki sumber daya yang akan diakses dan akun terpercaya berisi pengguna yang membutuhkan akses ke sumber daya. Namun, akun lain bisa saja memiliki sumber daya di akun Anda. Misalnya, akun kepercayaan mungkin mengizinkan akun tepercaya untuk membuat sumber daya baru, seperti membuat objek baru di bucket Amazon S3. Dalam hal ini, akun yang menciptakan sumber daya memiliki sumber daya dan mengendalikan siapa yang dapat mengakses sumber daya tersebut.

Setelah Anda membuat hubungan kepercayaan, IAM pengguna atau aplikasi dari akun tepercaya dapat menggunakan AWS Security Token Service (AWS STS) `AssumeRole`APIoperasi. Operasi ini menyediakan kredensi keamanan sementara yang memungkinkan akses ke AWS sumber daya di akun Anda.

Akun-akun tersebut dapat dikendalikan oleh Anda, atau akun dengan pengguna-pengguna dapat dikendalikan oleh pihak ketiga. Jika akun lain dengan pengguna adalah Akun AWS bahwa Anda tidak mengontrol, maka Anda dapat menggunakan `externalId` atribut. ID eksternal dapat berupa kata atau nomor apa pun yang disepakati antara Anda dan administrator akun pihak ketiga. Opsi ini secara otomatis menambahkan persyaratan kepada kebijakan kepercayaan yang memperbolehkan pengguna untuk mengasumsikan peran hanya jika permintaan mencakup `sts:ExternalID` yang benar. Untuk informasi selengkapnya, lihat Akses ke Akun AWS yang dimiliki oleh pihak ketiga.

Untuk informasi tentang cara menggunakan peran untuk mendelegasikan izin, lihat Istilah dan konsep peran. Untuk informasi tentang menggunakan peran layanan untuk memperbolehkan layanan mengakses sumber daya di akun Anda, lihat Membuat peran untuk mendelegasikan izin ke layanan AWS.

Membuat peran IAM (konsol)

Anda dapat menggunakan AWS Management Console untuk membuat peran yang dapat diasumsikan oleh IAM pengguna. Misalnya, asumsikan bahwa organisasi Anda memiliki beberapa Akun AWS untuk mengisolasi lingkungan pembangunan dari lingkungan produksi. Untuk informasi tingkat tinggi tentang membuat peran yang memungkinkan pengguna di akun pengembangan mengakses sumber daya di akun produksi, lihatContoh skenario menggunakan akun pengembangan dan produksi terpisah.

Izin minimum

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya IAM izin berikut:

- `access-analyzer:ValidatePolicy`
- `iam:AttachRolePolicy`
- `iam:CreatePolicy`
- `iam:CreateRole`
- `iam:GetAccountSummary`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:ListAccountAliases`
- `iam:ListAttachedRolePolicies`
- `iam:ListOpenIDConnectProviders`
- `iam:ListPolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`
- `iam:ListRoleTags`
- `iam:ListSAMLProviders`

Untuk membuat peran (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Pilih Akun AWStipe peran.
4. Untuk membuat peran untuk akun Anda, pilih Akun ini. Untuk membuat peran untuk akun lain, pilih Lainnya Akun AWSdan masukkan ID Akun yang ingin Anda berikan akses ke sumber daya Anda.

   Administrator akun yang ditentukan dapat memberikan izin untuk mengambil peran ini kepada IAM pengguna mana pun di akun itu. Untuk melakukannya, administrator melampirkan kebijakan

kepada pengguna atau grup yang memberikan izin untuk tindakan `sts:AssumeRole`. Kebijakan itu harus menentukan peran ARN sebagai`Resource`.

5. Jika Anda memberikan izin kepada pengguna dari akun yang tidak Anda kendalikan, dan pengguna akan mengambil peran ini secara terprogram, pilih Memerlukan ID eksternal. ID eksternal dapat berupa kata atau nomor apa pun yang disepakati antara Anda dan administrator akun pihak ketiga. Opsi ini secara otomatis menambahkan persyaratan kepada kebijakan kepercayaan yang memperbolehkan pengguna untuk mengasumsikan peran hanya jika permintaan mencakup `sts:ExternalID` yang benar. Untuk informasi selengkapnya, lihat Akses ke Akun AWS yang dimiliki oleh pihak ketiga.

   > **Important**
   >
   > Memilih opsi ini membatasi akses ke peran hanya melalui AWS CLI, Alat untuk Windows PowerShell, atau AWS API. Hal ini karena Anda tidak dapat menggunakan AWS konsol untuk beralih ke peran yang memiliki `externalId` kondisi dalam kebijakan kepercayaannya. Namun, Anda dapat membuat akses semacam ini secara terprogram dengan menulis skrip atau aplikasi menggunakan yang relevan. SDK Untuk informasi selengkapnya dan contoh skrip, lihat Cara Mengaktifkan Akses Lintas Akun ke AWS Management Consoledi AWS Blog Keamanan.

6. Jika Anda ingin membatasi peran untuk pengguna yang masuk dengan otentikasi multi-faktor (MFA), pilih Memerlukan. MFA Ini menambahkan kondisi pada kebijakan kepercayaan peran yang memeriksa MFA proses masuk. Pengguna yang ingin mengambil peran harus masuk dengan kata sandi satu kali sementara dari MFA perangkat yang dikonfigurasi. Pengguna tanpa MFA otentikasi tidak dapat mengambil peran. Untuk informasi selengkapnyaMFA, lihat AWS Otentikasi multi-faktor di IAM
7. Pilih Berikutnya.
8. IAMtermasuk daftar AWS kebijakan terkelola dan terkelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk kebijakan izin atau pilih Buat kebijakan untuk membuka tab peramban baru dan membuat kebijakan baru dari awal. Untuk informasi selengkapnya, lihat Membuat IAM kebijakan. Setelah Anda membuat kebijakan, tutup tab tersebut dan kembali ke tab asli Anda. Centang kotak di samping kebijakan izin yang Anda inginkan untuk dimiliki oleh siapa pun yang memegang peran tersebut. Jika Anda lebih suka, Anda boleh tidak memilih kebijakan saat ini, kemudian melampirkan kebijakan kepada peran di lain waktu. Secara default, peran tidak memiliki izin.
9. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan.

Buka bagian Atur batasan izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. Pilih kebijakan yang akan digunakan untuk batas izin.

10. Pilih Berikutnya.
11. Untuk Nama peran, masukkan nama peran Anda. Nama peran harus unik dalam diri Anda Akun AWS. Ketika nama peran digunakan dalam kebijakan atau sebagai bagian dari sebuahARN, nama peran tersebut peka huruf besar/kecil. Saat nama peran muncul ke pelanggan di konsol, seperti selama proses masuk, nama peran tidak peka huruf besar/kecil. Karena berbagai entitas mungkin mereferensikan peran tersebut, Anda tidak dapat mengedit nama peran setelah peran tersebut dibuat.
12. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
13. Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan bagian izin untuk mengedit kasus penggunaan dan izin untuk peran tersebut. Anda akan dikembalikan ke halaman sebelumnya untuk melakukan pengeditan.
14. (Opsional) Tambahkan metadata ke dalam peran dengan melampirkan tanda sebagai pasangan nilai-kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
15. Tinjau peran dan kemudian pilih Buat peran.

> **Important**
>
> Ingatlah bahwa ini hanya setengah bagian pertama dari konfigurasi yang diperlukan. Anda juga harus memberikan izin kepada pengguna individu dalam akun terpercaya untuk beralih ke peran dalam konsol, atau mengasumsikan peran tersebut secara terprogram. Untuk informasi selengkapnya tentang langkah ini, lihat Berikan izin pengguna untuk beralih peran.

Menciptakan IAM peran (AWS CLI)

Menciptakan peran dari AWS CLI melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan AWS CLI Anda harus secara eksplisit melakukan setiap langkah sendiri. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran untuk akses lintas akun (AWS CLI)

1. Buat peran: aws iam create-role
2. Lampirkan kebijakan izin terkelola ke peran: aws iam attach-role-policy

   atau

   Buat kebijakan izin sebaris untuk peran tersebut: aws iam put-role-policy
3. (Opsional) Tambahkan atribut khusus ke peran tersebut dengan melampirkan tag: aws iam tag-role

   Untuk informasi selengkapnya, lihat Mengelola tag pada IAM peran (AWS CLI atau AWS API).
4. (Opsional) Tetapkan batas izin untuk peran: aws iam put-role-permissions-boundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah lanjutan AWS fitur.

Contoh berikut menunjukkan dua langkah pertama, dan yang paling umum untuk membuat peran lintas akun dalam lingkungan sederhana. Contoh ini memungkinkan setiap pengguna dalam akun `123456789012` untuk mengasumsikan peran dan melihat `example_bucket` bucket Amazon S3. Contoh ini juga mengasumsikan bahwa Anda menggunakan komputer klien yang menjalankan Windows, dan telah mengonfigurasi antarmuka baris perintah Anda kredensial akun dan Wilayah Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS Antarmuka Baris Perintah.

Dalam contoh ini, sertakan kebijakan kepercayaan berikut pada perintah pertama ketika Anda membuat peran tersebut. Kebijakan kepercayaan ini memungkinkan pengguna di `123456789012` akun untuk mengambil peran menggunakan `AssumeRole` operasi, tetapi hanya jika pengguna memberikan MFA otentikasi menggunakan `TokenCode` parameter `SerialNumber` dan. Untuk informasi lebih lanjut tentangMFA, lihatAWS Otentikasi multi-faktor di IAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
          "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
      }
```

```
  ]
}
```

> **Important**
>
> Jika `Principal` elemen Anda berisi ARN untuk IAM peran atau pengguna tertentu, maka elemen ARN tersebut akan diubah menjadi ID utama unik saat kebijakan disimpan. Hal ini membantu memitigasi risiko seseorang meningkatkan izin mereka dengan menghapus dan membuat kembali peran atau pengguna. Anda biasanya tidak melihat ID ini di konsol karena ada juga transformasi terbalik kembali ke ARN saat kebijakan kepercayaan ditampilkan. Namun, jika Anda menghapus peran atau pengguna, maka ID utama muncul di konsol karena AWS tidak bisa lagi memetakannya kembali ke fileARN. Oleh karena itu, jika Anda menghapus dan membuat ulang pengguna atau peran yang direferensikan dalam `Principal` elemen kebijakan kepercayaan, Anda harus mengedit peran tersebut untuk mengganti peran tersebut. ARN

Saat menggunakan perintah kedua, Anda harus melampirkan kebijakan terkelola yang ada pada peran tersebut. Kebijakan izin berikut ini memperbolehkan siapa pun yang mengasumsikan peran untuk hanya melaksanakan tindakan `ListBucket` pada `example_bucket` bucket Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": "s3:ListBucket",
          "Resource": "arn:aws:s3:::example_bucket"
      }
  ]
}
```

Untuk membuat peran `Test-UserAccess-Role` ini, Anda harus terlebih dahulu menyimpan kebijakan kepercayaan sebelumnya dengan nama `trustpolicyforacct123456789012.json` ke folder `policies` di drive lokal `C:` Anda. Kemudian simpan kebijakan izin sebelumnya sebagai kebijakan terkelola pelanggan di Akun AWS dengan nama`PolicyForRole`. Kemudian Anda dapat menggunakan perintah berikut untuk membuat peran dan melampirkan kebijakan terkelola.

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> Ingatlah bahwa ini hanya setengah bagian pertama dari konfigurasi yang diperlukan. Anda juga harus memberikan izin kepada pengguna individu dalam akun terpercaya untuk beralih ke peran. Untuk informasi selengkapnya tentang langkah ini, lihat Berikan izin pengguna untuk beralih peran.

Setelah Anda membuat peran dan memberikan izin untuk melakukan AWS tugas atau akses AWS sumber daya, setiap pengguna di 123456789012 akun dapat mengambil peran. Untuk informasi selengkapnya, lihat Beralih ke IAM peran (AWS CLI).

Menciptakan IAM peran (AWS API)

Menciptakan peran dari AWS APImelibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan itu API Anda harus secara eksplisit melakukan setiap langkah sendiri. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran dalam kode (AWS API)

1. Buat peran: CreateRole

   Untuk kebijakan kepercayaan peran, Anda dapat menentukan lokasi file.
2. Lampirkan kebijakan izin terkelola ke peran: AttachRolePolicy

   atau

   Membuat kebijakan izin inline untuk peran tersebut: PutRolePolicy

> **Important**
>
> Ingatlah bahwa ini hanya setengah bagian pertama dari konfigurasi yang diperlukan. Anda juga harus memberikan izin kepada pengguna individu dalam akun terpercaya untuk beralih ke peran. Untuk informasi selengkapnya tentang langkah ini, lihat Berikan izin pengguna untuk beralih peran.

3. (Opsional) Tambahkan atribut khusus ke pengguna dengan melampirkan tag: TagRole

   Untuk informasi selengkapnya, lihat Mengelola tag pada IAM pengguna (AWS CLI atau AWS API).

4. (Opsional) Tetapkan batas izin untuk peran: PutRolePermissionsBoundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah lanjutan AWS fitur.

Setelah Anda membuat peran dan memberikan izin untuk melakukan AWS tugas atau akses AWS sumber daya, Anda harus memberikan izin kepada pengguna di akun untuk memungkinkan mereka mengambil peran. Untuk informasi lebih lanjut tentang mengambil peran, lihat Beralih ke IAM peran (AWS API).

Menciptakan IAM peran (AWS CloudFormation)

Untuk informasi tentang membuat IAM peran di AWS CloudFormation, lihat referensi sumber daya dan properti dan contoh di AWS CloudFormation Panduan Pengguna.

Untuk informasi lebih lanjut tentang IAM template di AWS CloudFormation, lihat AWS Identity and Access Management cuplikan template di AWS CloudFormation Panduan Pengguna.

Membuat peran untuk mendelegasikan izin ke layanan AWS

Banyak AWS layanan mengharuskan Anda menggunakan peran untuk memungkinkan layanan mengakses sumber daya di layanan lain atas nama Anda. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut peran layanan. Ketika suatu peran melayani tujuan khusus untuk suatu layanan, itu dikategorikan sebagai peran terkait layanan. Untuk melihat apa yang didukung layanan dengan menggunakan peran yang terkait dengan layanan, atau apakah layanan mendukung segala bentuk kredensial sementara, lihat AWS layanan yang bekerja dengan IAM.

Untuk mempelajari bagaimana layanan individu menggunakan peran, pilih nama layanan dalam tabel untuk melihat dokumentasi untuk layanan tersebut.

Saat menyetel `PassRole` izin, Anda harus memastikan bahwa pengguna tidak melewati peran di mana peran tersebut memiliki lebih banyak izin daripada yang Anda inginkan untuk dimiliki pengguna. Misalnya, Alice mungkin tidak diizinkan untuk melakukan tindakan Amazon S3 apa pun. Jika Alice dapat meneruskan peran ke layanan yang memungkinkan tindakan Amazon S3, layanan dapat melakukan tindakan Amazon S3 atas nama Alice saat menjalankan pekerjaan.

Untuk informasi tentang cara peran membantu Anda untuk mendelegasikan izin, lihat Istilah dan konsep peran.

Izin peran layanan

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) membuat atau mengedit peran layanan.

> **Note**
>
> ARNUntuk peran terkait layanan mencakup prinsip layanan, yang ditunjukkan dalam kebijakan berikut sebagai. *`SERVICE-NAME`*`.amazonaws.com` Jangan mencoba menebak prinsip layanan, karena peka huruf besar/kecil dan formatnya dapat bervariasi antar AWS layanan. Untuk melihat prinsipal layanan untuk suatu layanan, lihat dokumentasi peran yang terkait dengan layanan.

Untuk memungkinkan IAM entitas membuat peran layanan tertentu

Tambahkan kebijakan berikut ke IAM entitas yang perlu membuat peran layanan. Kebijakan ini memungkinkan Anda untuk membuat peran layanan untuk layanan tertentu dan dengan nama yang spesifik. Anda dapat melampirkan kebijakan terkelola atau inline pada peran tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
```

```
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
}
```

Untuk memungkinkan IAM entitas membuat peran layanan apa pun

AWS merekomendasikan bahwa Anda hanya mengizinkan pengguna administratif untuk membuat peran layanan apa pun. Seseorang yang memiliki izin untuk membuat peran dan melampirkan kebijakan apa pun dapat meningkatkan izinnya sendiri. Sebagai gantinya, buat kebijakan yang memungkinkan mereka hanya membuat peran yang mereka butuhkan atau minta administrator membuat peran layanan atas nama mereka.

Untuk melampirkan kebijakan yang memungkinkan administrator mengakses seluruh kebijakan Anda Akun AWS, gunakan kebijakan AdministratorAccess AWS terkelola.

Untuk mengizinkan IAM entitas mengedit peran layanan

Tambahkan kebijakan berikut ke IAM entitas yang perlu mengedit peran layanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
```

```
            "Sid": "ViewRolesAndPolicies",
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk mengizinkan IAM entitas menghapus peran layanan tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menghapus peran layanan yang ditentukan.

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

Untuk mengizinkan IAM entitas menghapus peran layanan apa pun

AWS merekomendasikan bahwa Anda hanya mengizinkan pengguna administratif untuk menghapus peran layanan apa pun. Sebagai gantinya, buat kebijakan yang memungkinkan mereka menghapus hanya peran yang mereka butuhkan atau minta administrator menghapus peran layanan atas nama mereka.

Untuk melampirkan kebijakan yang memungkinkan administrator mengakses seluruh kebijakan Anda Akun AWS, gunakan kebijakan AdministratorAccess AWS terkelola.

Membuat peran untuk AWS layanan (konsol)

Anda dapat menggunakan AWS Management Console untuk membuat peran untuk layanan. Karena beberapa layanan mendukung lebih dari satu peran layanan, lihat dokumentasi AWS agar layanan Anda dapat melihat kasus penggunaan mana yang dapat dipilih. Anda dapat mempelajari cara menetapkan kebijakan kepercayaan dan izin yang diperlukan untuk peran tersebut sehingga layanan dapat mengasumsikan peran tersebut atas nama Anda. Langkah-langkah yang dapat Anda gunakan untuk mengontrol izin untuk peran Anda dapat bervariasi, tergantung pada bagaimana layanan mendefinisikan kasus penggunaan, dan apakah Anda membuat peran terkait layanan atau tidak.

Untuk membuat peran untuk Layanan AWS (IAMkonsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi IAM konsol, pilih Peran, lalu pilih Buat peran.
3. Untuk jenis entitas Tepercaya, pilih Layanan AWS.
4. Untuk kasus Layanan atau penggunaan, pilih layanan, lalu pilih kasus penggunaan. Kasus penggunaan ditentukan oleh layanan untuk menyertakan kebijakan kepercayaan yang diperlukan layanan.
5. Pilih Berikutnya.
6. Untuk kebijakan Izin, opsi bergantung pada kasus penggunaan yang Anda pilih:
   - Jika layanan menentukan izin untuk peran tersebut, Anda tidak dapat memilih kebijakan izin.
   - Pilih dari serangkaian kebijakan izin terbatas.
   - Pilih dari semua kebijakan izin.
   - Pilih kebijakan tanpa izin, buat kebijakan setelah peran dibuat, lalu lampirkan kebijakan ke peran.
7. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.
   a. Buka bagian Setel batas izin, lalu pilih Gunakan batas izin untuk mengontrol izin peran maksimum.

      IAMmenyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda.
   b. Pilih kebijakan yang akan digunakan untuk batas izin.
8. Pilih Berikutnya.
9. Untuk nama Peran, opsi bergantung pada layanan:
   - Jika layanan menentukan nama peran, Anda tidak dapat mengedit nama peran.
   - Jika layanan mendefinisikan awalan untuk nama peran, Anda dapat memasukkan akhiran opsional.
   - Jika layanan tidak menentukan nama peran, Anda dapat memberi nama peran.

   > **Important**
   >
   > Saat Anda memberi nama peran, perhatikan hal berikut:

- Nama peran harus unik di dalam diri Anda Akun AWS, dan tidak dapat dibuat unik berdasarkan kasus.

  Misalnya, jangan membuat peran bernama keduanya **PRODROLE** dan**prodrole**. Ketika nama peran digunakan dalam kebijakan atau sebagai bagian dariARN, nama peran akan peka huruf besar/kecil, namun ketika nama peran muncul kepada pelanggan di konsol, seperti selama proses login, nama peran tersebut tidak peka huruf besar/kecil.

- Anda tidak dapat mengedit nama peran setelah dibuat karena entitas lain mungkin mereferensikan peran tersebut.

10. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran tersebut.
11. (Opsional) Untuk mengedit kasus penggunaan dan izin untuk peran, di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin, pilih Edit.
12. (Opsional) Untuk membantu mengidentifikasi, mengatur, atau mencari peran, tambahkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tagIAM, lihat Menandai IAM sumber daya di Panduan IAM Pengguna.
13. Tinjau peran lalu pilih Buat peran.

Membuat peran untuk layanan (AWS CLI)

Membuat peran dari AWS CLI melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan itu AWS CLI Anda harus secara eksplisit melakukan setiap langkah sendiri. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Jika layanan yang Anda kerjakan adalah AmazonEC2, maka Anda juga harus membuat profil instance dan menambahkan peran ke dalamnya. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran untuk AWS layanan dari AWS CLI

1. `create-role`Perintah berikut membuat peran bernama Test-Role dan melampirkan kebijakan kepercayaan padanya:

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```

2. Lampirkan kebijakan izin terkelola ke peran: aws attach-role-policy iam.

Misalnya, `attach-role-policy` perintah berikut melampirkan kebijakan AWS terkelola bernama `ReadOnlyAccess` ke IAM peran bernama`ReadOnlyRole`:

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
ReadOnlyAccess --role-name ReadOnlyRole
```

atau

Buat kebijakan izin sebaris untuk peran tersebut: aws iam put-role-policy

Untuk menambahkan kebijakan izin sebaris, lihat contoh berikut:

```
aws iam put-role-policy --role-name Test-Role --policy-name
ExamplePolicy --policy-document file://AdminPolicy.json
```

3. (Opsional) Tambahkan atribut khusus ke peran tersebut dengan melampirkan tag: aws iam tag-role

   Untuk informasi selengkapnya, lihat Mengelola tag pada IAM peran (AWS CLI atau AWS API).

4. (Opsional) Tetapkan batas izin untuk peran: aws iam put-role-permissions-boundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Jika Anda akan menggunakan peran dengan Amazon EC2 atau AWS layanan lain yang menggunakan AmazonEC2, Anda harus menyimpan peran tersebut dalam profil instance. Profil instance adalah wadah untuk peran yang dapat dilampirkan ke EC2 instance Amazon saat diluncurkan. Profil instans hanya dapat berisi satu , dan batas tersebut tidak dapat ditingkatkan. Jika Anda membuat peran menggunakan AWS Management Console, profil instance akan dibuat untuk Anda dengan nama yang sama dengan peran. Untuk informasi selengkapnya tentang profil instans, lihat Gunakan profil contoh. Untuk informasi tentang cara meluncurkan EC2 instance dengan peran, lihat Mengontrol Akses ke EC2 Sumber Daya Amazon di Panduan EC2 Pengguna Amazon.

Untuk membuat profil instans dan menyimpan peran di dalamnya (AWS CLI)

1. Buat profil instance: aws iam create-instance-profile
2. Tambahkan peran ke profil instance: aws iam add-role-to-instance -profile

Perintah AWS CLI contoh yang ditetapkan di bawah ini menunjukkan dua langkah pertama untuk membuat peran dan melampirkan izin. Itu juga menunjukkan dua langkah untuk membuat profil instans dan menambahkan peran ke profil. Contoh kebijakan kepercayaan ini memungkinkan EC2 layanan Amazon untuk mengambil peran dan melihat bucket `example_bucket` Amazon S3. Contoh ini juga mengasumsikan bahwa Anda sedang bekerja di komputer klien yang menjalankan Windows dan telah mengonfigurasi antarmuka baris perintah Anda dengan kredensial akun dan Wilayah Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi Antarmuka Baris AWS Perintah.

Dalam contoh ini, sertakan kebijakan kepercayaan berikut pada perintah pertama ketika Anda membuat peran tersebut. Kebijakan kepercayaan ini memungkinkan EC2 layanan Amazon untuk mengambil peran.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

Saat Anda menggunakan perintah kedua, Anda harus melampirkan kebijakan izin pada peran tersebut. Contoh kebijakan izin berikut memungkinkan peran untuk hanya mengasumsikan tindakan `ListBucket` pada `example_bucket` bucket Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Untuk membuat peran ini `Test-Role-for-EC2`, Anda harus terlebih dahulu menyimpan kebijakan kepercayaan sebelumnya dengan nama `trustpolicyforec2.json` dan kebijakan izin sebelumnya dengan nama `permissionspolicyforec2.json` ke `policies` direktori di lokasi Anda `C:`. Anda dapat menggunakan perintah berikut untuk membuat peran, melampirkan kebijakan, membuat profil instans, dan menambahkan peran ke profil instans.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json

# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

Saat meluncurkan EC2 instance, tentukan nama profil instance di halaman Konfigurasikan Detail Instance jika Anda menggunakan AWS konsol. Jika Anda menggunakan `aws ec2 run-instances` CLI perintah, tentukan `--iam-instance-profile` parameternya.

Membuat peran untuk layanan (AWS API)

Membuat peran dari AWS API melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan itu API Anda harus secara eksplisit melakukan setiap langkah sendiri. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Jika layanan yang Anda kerjakan adalah AmazonEC2, maka Anda juga harus membuat profil instance dan menambahkan peran ke dalamnya. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran untuk AWS layanan (AWS API)

1. Buat peran: CreateRole

   Untuk kebijakan kepercayaan peran, Anda dapat menentukan lokasi file.
2. Lampirkan kebijakan izin terkelola ke peran: AttachRolePolicy

   atau

   Membuat kebijakan izin sebaris untuk peran tersebut: PutRolePolicy
3. (Opsional) Tambahkan atribut khusus ke pengguna dengan melampirkan tag: TagRole

Untuk informasi selengkapnya, lihat Mengelola tag pada IAM pengguna (AWS CLI atau AWS API).

4. (Opsional) Tetapkan batas izin untuk peran: PutRolePermissionsBoundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Jika Anda akan menggunakan peran dengan Amazon EC2 atau AWS layanan lain yang menggunakan AmazonEC2, Anda harus menyimpan peran tersebut dalam profil instance. Profil contoh adalah wadah untuk peran. Profil instans hanya dapat berisi satu peran, dan batas tersebut tidak dapat ditingkatkan. Jika Anda membuat peran di AWS Management Console, profil instans dibuat untuk Anda dengan nama yang sama dengan peran. Untuk informasi selengkapnya tentang profil instans, lihat Gunakan profil contoh. Untuk informasi tentang cara meluncurkan EC2 instans Amazon dengan peran, lihat Mengontrol Akses ke EC2 Sumber Daya Amazon di Panduan EC2 Pengguna Amazon.

Untuk membuat profil instance dan menyimpan peran di dalamnya (AWS API)

1. Buat profil instans: CreateInstanceProfile
2. Tambahkan peran ke profil instance: AddRoleToInstanceProfile

Buat peran tertaut layanan

Peran terkait layanan adalah jenis peran unik yang IAM ditautkan langsung ke layanan. AWS Peran terkait layanan telah ditentukan sebelumnya oleh layanan dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda. Layanan yang terhubung juga menentukan cara Anda membuat, memodifikasi, dan menghapus peran yang terkait dengan layanan. Layanan dapat secara otomatis membuat atau menghapus peran. Peran ini memungkinkan Anda membuat, memodifikasi, atau menghapus peran sebagai bagian dari wizard atau proses dalam layanan. Atau mungkin mengharuskan Anda menggunakan IAM untuk membuat atau menghapus peran. Terlepas dari metodenya, peran terkait layanan menyederhanakan proses penyiapan layanan karena Anda tidak perlu menambahkan izin secara manual untuk layanan untuk menyelesaikan tindakan atas nama Anda.

> **Note**
>
> Ingatlah bahwa peran layanan berbeda dari peran terkait layanan. Peran layanan adalah IAMperan yang diasumsikan layanan untuk melakukan tindakan atas nama Anda. IAMAdministrator dapat membuat, memodifikasi, dan menghapus peran layanan dari dalamIAM. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke Layanan AWS dalam IAMPanduan Pengguna. Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. IAMAdministrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.

Layanan tertaut menentukan izin peran tertaut-layanannya, dan kecuali ditentukan lain, hanya layanan itu yang dapat mengambil peran tersebut. Izin yang ditetapkan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas lain mana pun. IAM

Sebelum Anda dapat menghapus peran, Anda harus terlebih dahulu menghapus sumber daya terkait mereka. Ini melindungi sumber daya Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

> **Tip**
>
> Untuk informasi tentang layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait-Layanan Pilih Ya dengan tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan

Anda harus mengonfigurasi izin untuk IAM entitas (pengguna atau peran) agar pengguna atau peran dapat membuat atau mengedit peran terkait layanan.

> **Note**
>
> Peran ARN untuk layanan terkait mencakup prinsip layanan, yang ditunjukkan dalam kebijakan di bawah ini sebagai. `SERVICE-NAME.amazonaws.com` Jangan mencoba

menebak prinsipal layanan, karena peka huruf besar/kecil dan formatnya dapat bervariasi antar AWS layanan. Untuk melihat prinsipal layanan untuk suatu layanan, lihat dokumentasi peran yang terkait dengan layanan.

Untuk mengizinkan IAM entitas membuat peran terkait layanan tertentu

Tambahkan kebijakan berikut ke IAM entitas yang perlu membuat peran terkait layanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

Untuk mengizinkan IAM entitas membuat peran terkait layanan

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu membuat peran terkait layanan, atau peran layanan apa pun yang menyertakan kebijakan yang diperlukan. Pernyataan kebijakan ini tidak mengizinkan IAM entitas untuk melampirkan kebijakan ke peran tersebut.

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
```

```
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Untuk mengizinkan IAM entitas mengedit deskripsi peran layanan apa pun

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu mengedit deskripsi peran terkait layanan, atau peran layanan apa pun.

```
{
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Untuk mengizinkan IAM entitas menghapus peran terkait layanan tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menghapus peran terkait layanan.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

Untuk mengizinkan IAM entitas menghapus peran terkait layanan

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menghapus peran terkait layanan, tetapi bukan peran layanan.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
```

```
}
```

Untuk memungkinkan IAM entitas meneruskan peran yang ada ke layanan

Beberapa AWS layanan memungkinkan Anda untuk meneruskan peran yang ada ke layanan, alih-alih membuat peran terkait layanan baru. Untuk melakukannya, pengguna harus memiliki izin untuk melewati peran tersebut dengan layanan tersebut. Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu meneruskan peran. Pernyataan kebijakan ini juga memungkinkan entitas untuk melihat daftar peran yang darinya mereka dapat memilih peran untuk diteruskan. Untuk informasi selengkapnya, lihat Berikan izin pengguna untuk meneruskan peran ke layanan AWS.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

Izin tidak langsung dengan peran terkait layanan

Izin yang diberikan oleh peran terkait layanan dapat ditransfer secara tidak langsung ke pengguna dan peran lain. Ketika peran terkait layanan digunakan oleh AWS layanan, peran terkait layanan tersebut dapat menggunakan izinnya sendiri untuk memanggil layanan lain. AWS Ini berarti bahwa pengguna dan peran dengan izin untuk memanggil layanan yang menggunakan peran terkait layanan mungkin memiliki akses tidak langsung ke layanan yang dapat diakses oleh peran terkait layanan tersebut.

Misalnya, saat Anda membuat instans Amazon RDS DB, peran yang ditautkan layanan akan RDS dibuat secara otomatis jika belum ada. Peran terkait layanan ini memungkinkan Anda RDS untuk memanggil Amazon, EC2 Amazon, SNS Amazon CloudWatch Logs, dan Amazon Kinesis atas nama Anda. Jika Anda mengizinkan pengguna dan peran di akun Anda untuk memodifikasi atau membuat RDS database, mereka mungkin dapat berinteraksi secara tidak langsung dengan Amazon, Amazon, CloudWatch log EC2 SNS Amazon Logs, dan sumber daya Amazon Kinesis dengan RDS

menelepon, RDS seperti yang akan menggunakan peran terkait layanan untuk mengakses sumber daya tersebut.

Membuat peran terkait layanan

Metode yang Anda gunakan untuk membuat peran terkait layanan tergantung pada layanannya. Dalam beberapa kasus, Anda tidak perlu membuat peran terkait layanan secara manual. Misalnya, ketika Anda menyelesaikan tindakan tertentu (seperti membuat sumber daya) dalam layanan, layanan tersebut mungkin membuat peran yang berkaitan dengan layanan bagi Anda. Atau jika Anda menggunakan layanan sebelum mulai mendukung peran tertaut-layanan, maka layanan tersebut mungkin secara otomatis membuat peran tersebut di akun Anda. Untuk mempelajari informasi selengkapnya, lihat Peran baru muncul di akun AWS saya.

Dalam kasus lain, layanan mungkin mendukung pembuatan peran terkait layanan secara manual menggunakan konsol layananAPI, atau. CLI Untuk informasi tentang layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait-Layanan Untuk mempelajari apakah layanan mendukung pembuatan peran tertaut-layanan, pilih tautan Ya untuk melihat dokumentasi peran tertaut-layanan untuk layanan itu.

Jika layanan tidak mendukung pembuatan peran, maka Anda dapat menggunakannya IAM untuk membuat peran terkait layanan.

> **Important**
>
> Peran terkait layanan diperhitungkan terhadap IAMperan Anda dalam Akun AWS batas, tetapi jika Anda telah mencapai batas, Anda masih dapat membuat peran terkait layanan di akun Anda. Hanya peran yang berkaitan dengan layanan yang dapat melebihi batas.

Membuat peran terkait layanan (konsol)

Sebelum membuat peran terkait layananIAM, cari tahu apakah layanan tertaut secara otomatis membuat peran terkait layanan, Selain itu, pelajari apakah Anda dapat membuat peran dari konsol layanan, atau. API CLI

Untuk membuat peran terkait layanan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di panel navigasi IAM konsol, pilih Peran. Kemudian, pilih Buat peran.
3. Pilih jenis peran AWS Layanan.
4. Pilih kasus penggunaan untuk layanan Anda. Kasus penggunaan ditentukan oleh layanan untuk menyertakan kebijakan kepercayaan yang disyaratkan oleh layanan. Lalu, pilih Selanjutnya.
5. Pilih satu kebijakan izin atau lebih untuk dilampirkan ke peran tersebut. Bergantung pada kasus penggunaan yang Anda pilih, layanan mungkin melakukan salah satu hal berikut:

   - Tentukan izin yang digunakan oleh peran.
   - Memungkinkan Anda memilih dari serangkaian izin terbatas.
   - Memungkinkan Anda memilih dari izin apa pun.
   - Memungkinkan Anda memilih tidak ada kebijakan saat ini, membuat kebijakan nanti, lalu melampirkannya ke peran.

   Pilih kotak centang di samping kebijakan yang menetapkan izin yang Anda inginkan untuk peran tersebut, lalu pilih Berikutnya.

   > **Note**
   >
   > Izin yang Anda tentukan tersedia untuk setiap entitas yang menggunakan peran tersebut. Secara default, peran tidak memiliki izin.

6. Untuk Nama peran, tingkat penyesuaian nama peran ditentukan oleh layanan. Jika layanan mendefinisikan nama peran, maka opsi ini tidak dapat diedit. Dalam kasus lain, layanan mungkin menentukan awalan untuk peran dan memungkinkan Anda memasukkan akhiran opsional.

   Jika memungkinkan, masukkan akhiran nama peran untuk ditambahkan ke nama default. Akhiran ini membantu Anda mengidentifikasi tujuan peran ini. Nama peran harus unik di akun AWS Anda. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat peran dengan nama **`<service-linked-role-name>_SAMPLE`** dan **`<service-linked-role-name>_sample`**. Anda tidak dapat mengubah nama peran setelah dibuat karena berbagai entitas mungkin mereferensikan peran tersebut.
7. (Opsional) Untuk Deskripsi, edit deskripsi untuk peran terkait layanan baru.
8. Anda tidak dapat melampirkan tanda ke peran terkait layanan selama pembuatan. Untuk informasi selengkapnya tentang penggunaan tag dIAM, lihatTag untuk AWS Identity and Access Management sumber daya.

9. Tinjau peran dan kemudian pilih Buat peran.

Membuat peran terkait layanan (AWS CLI)

Sebelum membuat peran terkait layananIAM, cari tahu apakah layanan tertaut secara otomatis membuat peran terkait layanan dan apakah Anda dapat membuat peran dari layanan. CLI Jika layanan tidak CLI didukung, Anda dapat menggunakan IAM perintah untuk membuat peran terkait layanan dengan kebijakan kepercayaan dan kebijakan sebaris yang diperlukan layanan untuk mengambil peran tersebut.

Untuk membuat peran terkait layanan ()AWS CLI

Jalankan perintah berikut:

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Membuat peran terkait layanan ()AWS API

Sebelum membuat peran terkait layananIAM, cari tahu apakah layanan tertaut secara otomatis membuat peran terkait layanan dan apakah Anda dapat membuat peran dari layanan. API Jika layanan tidak API didukung, Anda dapat menggunakannya AWS API untuk membuat peran terkait layanan dengan kebijakan kepercayaan dan kebijakan sebaris yang diperlukan layanan untuk mengambil peran tersebut.

Untuk membuat peran terkait layanan ()AWS API

Gunakan CreateServiceLinkedRoleAPIpanggilan. Dalam permintaan, sebutkan nama layanan dari *SERVICE_NAME_URL*`.amazonaws.com`.

Misalnya, untuk membuat peran terkait layanan Lex Bots, gunakan `lex.amazonaws.com`.

Buat peran untuk penyedia identitas pihak ketiga (federasi)

Anda dapat menggunakan penyedia identitas alih-alih membuat IAM pengguna di situs Anda Akun AWS. Dengan penyedia identitas (iDP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk mengakses AWS sumber daya di akun Anda. Untuk informasi lebih lanjut tentang federasi dan penyedia identitas, lihat Penyedia dan federasi identitas.

Membuat peran untuk pengguna federasi (konsol)

Prosedur untuk membuat peran bagi pengguna federasi bergantung pada pilihan penyedia pihak ketiga Anda:

- Untuk OpenID Connect (OIDC), lihat. Buat peran untuk federasi OpenID Connect (konsol)
- Untuk SAML 2.0, lihatBuat peran untuk federasi SAML 2.0 (konsol).

Membuat peran untuk akses gabungan (AWS CLI)

Langkah-langkah untuk membuat peran untuk penyedia identitas yang didukung (OIDCatauSAML) dari AWS CLI yang sama. Perbedaannya ada dalam konten kebijakan kepercayaan yang Anda buat dalam langkah-langkah prasyarat. Mulai dengan mengikuti langkah-langkah dalam bagian Prasyarat untuk jenis penyedia yang Anda gunakan:

- Untuk OIDC penyedia, lihatPrasyarat untuk menciptakan peran untuk OIDC.
- Untuk SAML penyedia, lihatPrasyarat untuk menciptakan peran untuk SAML.

Membuat peran dari AWS CLI melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan itu AWS CLI Anda harus secara eksplisit melakukan setiap langkah sendiri. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran bagi federasi identitas (AWS CLI)

1. Buat peran: aws iam create-role
2. Lampirkan kebijakan izin ke peran: aws iam attach-role-policy

   atau

   Buat kebijakan izin sebaris untuk peran tersebut: aws iam put-role-policy
3. (Opsional) Tambahkan atribut khusus ke peran tersebut dengan melampirkan tag: aws iam tag-role

   Untuk informasi selengkapnya, lihat Mengelola tag pada IAM peran (AWS CLI atau AWS API).
4. (Opsional) Tetapkan batas izin untuk peran: aws iam put-role-permissions-boundary

Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Contoh berikut menunjukkan dua langkah pertama, dan paling umum, untuk membuat peran penyedia identitas dalam lingkungan yang sederhana. Contoh ini memungkinkan setiap pengguna dalam akun `123456789012` untuk mengasumsikan peran dan melihat `example_bucket` bucket Amazon S3. Contoh ini juga mengasumsikan bahwa Anda menjalankan AWS CLI pada komputer yang menjalankan Windows, dan telah mengkonfigurasi AWS CLI dengan kredensi Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS Command Line Interface.

Contoh kebijakan kepercayaan berikut ini dirancang untuk aplikasi seluler jika pengguna masuk dengan menggunakan Amazon Cognito. Dalam contoh ini, *`us-east:12345678-ffff-ffff-ffff-123456`* mewakili ID kumpulan identitas yang ditetapkan oleh Amazon Cognito.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

Kebijakan izin berikut ini memperbolehkan siapa pun yang mengasumsikan peran untuk hanya melaksanakan tindakan `ListBucket` pada `example_bucket` bucket Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Untuk membuat peran ini `Test-Cognito-Role`, Anda harus terlebih dahulu menyimpan kebijakan kepercayaan sebelumnya dengan nama `trustpolicyforcognitofederation.json` dan kebijakan izin sebelumnya dengan nama `permspolicyforcognitofederation.json` ke `policies` direktori di lokasi Anda `C:`. Kemudian Anda dapat menggunakan perintah berikut untuk membuat peran dan melampirkan kebijakan inline.

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json

# Attach the permissions policy to the role to specify what it is allowed to do.
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

Membuat peran untuk akses federasi ()AWS API

Langkah-langkah untuk membuat peran untuk penyedia identitas yang didukung (OIDCatauSAML) dari AWS CLI yang sama. Perbedaannya ada dalam konten kebijakan kepercayaan yang Anda buat dalam langkah-langkah prasyarat. Mulai dengan mengikuti langkah-langkah dalam bagian Prasyarat untuk jenis penyedia yang Anda gunakan:

- Untuk OIDC penyedia, lihatPrasyarat untuk menciptakan peran untuk OIDC.
- Untuk SAML penyedia, lihatPrasyarat untuk menciptakan peran untuk SAML.

Untuk membuat peran federasi identitas (AWS API)

1. Buat peran: CreateRole
2. Lampirkan kebijakan izin ke peran: AttachRolePolicy

   atau

   Membuat kebijakan izin sebaris untuk peran tersebut: PutRolePolicy
3. (Opsional) Tambahkan atribut khusus ke pengguna dengan melampirkan tag: TagRole

   Untuk informasi selengkapnya, lihat Mengelola tag pada IAM pengguna (AWS CLI atau AWS API).
4. (Opsional) Tetapkan batas izin untuk peran: PutRolePermissionsBoundary

Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Buat peran untuk federasi OpenID Connect (konsol)

Anda dapat menggunakan penyedia identitas federasi OpenID Connect (OIDC) alih-alih membuat AWS Identity and Access Management pengguna di penyedia identitas Anda. Akun AWS Dengan penyedia identitas (iDP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk mengakses AWS sumber daya di akun Anda. Untuk informasi lebih lanjut tentang federasi dan IdPs, lihatPenyedia dan federasi identitas.

Prasyarat untuk menciptakan peran untuk OIDC

Sebelum Anda dapat membuat peran untuk OIDC federasi, Anda harus terlebih dahulu menyelesaikan langkah-langkah prasyarat berikut.

Untuk mempersiapkan diri untuk menciptakan peran untuk OIDC federasi

1. Daftar dengan satu atau lebih layanan yang menawarkan OIDC identitas federasi. Jika Anda membuat aplikasi yang memerlukan akses ke AWS sumber daya, Anda juga mengonfigurasi aplikasi dengan informasi penyedia. Saat melakukannya, penyedia memberi Anda aplikasi atau ID audiens yang unik untuk aplikasi Anda. (Penyedia berbeda menggunakan terminologi berbeda untuk proses ini. Panduan ini menggunakan istilah konfigurasi untuk proses mengidentifikasi aplikasi Anda dengan penyedia.) Anda dapat mengonfigurasi beberapa aplikasi dengan setiap penyedia, atau beberapa penyedia dengan satu aplikasi. Lihat informasi tentang penggunaan penyedia identitas sebagai berikut:

   - Login dengan Amazon Developer Center
   - Tambahkan Login Facebook ke Aplikasi atau Website Anda di situs pengembang Facebook.
   - Menggunakan OAuth 2.0 untuk Login (OpenID Connect) di situs pengembang Google.
2. Setelah Anda menerima informasi yang diperlukan dari iDP, buat iDP di. IAM Untuk informasi selengkapnya, lihat Buat penyedia identitas OpenID Connect (OIDC) di IAM.

> **Important**
>
> Jika Anda menggunakan OIDC iDP dari Google, Facebook, atau Amazon Cognito, jangan membuat IAM iDP terpisah di file. AWS Management Console Penyedia OIDC

identitas ini sudah dibangun AWS dan tersedia untuk Anda gunakan. Lewati langkah ini dan buat peran baru menggunakan IDP Anda di langkah berikut.

3. Persiapkan kebijakan untuk peran yang akan diaambil oleh pengguna yang diotentikasi IdP. Sebagaimana peran apa pun, peran untuk aplikasi seluler mencakup dua kebijakan. Salah satunya adalah kebijakan kepercayaan yang menentukan siapa yang dapat mengasumsikan peran tersebut. Lainnya adalah kebijakan izin yang menentukan tindakan AWS dan sumber daya yang diperbolehkan atau ditolak untuk diakses aplikasi seluler.

   Untuk web IdPs, kami menyarankan Anda menggunakan Amazon Cognito untuk mengelola identitas. Dalam hal ini, gunakan kebijakan kepercayaan yang serupa dengan contoh ini.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

   Ganti `us-east-2:12345678-abcd-abcd-abcd-123456` dengan ID kumpulan identitas yang diberikan Amazon Cognito kepada Anda.

   Jika Anda mengonfigurasi OIDC iDP secara manual, saat membuat kebijakan kepercayaan, Anda harus menggunakan tiga nilai yang memastikan bahwa hanya aplikasi yang dapat mengambil peran tersebut:

   - Untuk elemen `Action`, gunakan tindakan `sts:AssumeRoleWithWebIdentity`.
   - Untuk elemen `Principal`, gunakan string `{"Federated":`*`providerUrl/providerArn`*`}`.
     - Untuk beberapa umum OIDC IdPs, *`providerUrl`* adalah aURL. Contoh berikut termasuk metode untuk menentukan prinsipal untuk beberapa umum IdPs:

       `"Principal":{"Federated":"cognito-identity.amazonaws.com"}`

```
"Principal":{"Federated":"www.amazon.com"}

"Principal":{"Federated":"graph.facebook.com"}

"Principal":{"Federated":"accounts.google.com"}
```

- Untuk OIDC penyedia lain, gunakan Amazon Resource Name (ARN) dari OIDC idP yang Anda buatStep 2, seperti contoh berikut:

```
"Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
server.example.com"}
```

- Untuk elemen `Condition` , gunakan syarat `StringEquals` untuk membatasi izin. Uji ID kumpulan identitas untuk Amazon Cognito) atau ID aplikasi untuk penyedia lain. ID kumpulan identitas harus sesuai dengan ID aplikasi yang Anda terima saat mengonfigurasi aplikasi dengan iDP. Pencocokan antara IDs memastikan bahwa permintaan berasal dari aplikasi Anda.

> **Note**
>
> IAMperan untuk kumpulan identitas Amazon Cognito mempercayai prinsip layanan `cognito-identity.amazonaws.com` untuk mengambil peran tersebut. Peran jenis ini harus mengandung setidaknya satu kunci kondisi untuk membatasi kepala sekolah yang dapat mengambil peran tersebut.
> Pertimbangan tambahan berlaku untuk kumpulan identitas Amazon Cognito yang mengambil peran IAM lintas akun. Kebijakan kepercayaan dari peran ini harus menerima prinsip `cognito-identity.amazonaws.com` layanan dan harus berisi kunci `aud` kondisi untuk membatasi asumsi peran bagi pengguna dari kumpulan identitas yang Anda inginkan. Kebijakan yang mempercayai kumpulan identitas Amazon Cognito tanpa kondisi ini menimbulkan risiko bahwa pengguna dari kumpulan identitas yang tidak diinginkan dapat mengambil peran tersebut. Untuk informasi selengkapnya, lihat Kebijakan kepercayaan untuk IAM peran dalam autentikasi Dasar (Klasik) di Panduan Pengembang Amazon Cognito.

Buat elemen kondisi yang mirip dengan salah satu contoh berikut, tergantung pada idP yang Anda gunakan:

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
"us-east:12345678-ffff-ffff-ffff-123456"}}

"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}

"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}

"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

Untuk OIDC penyedia, gunakan OIDC IDP URL yang memenuhi syarat penuh dengan kunci aud konteks, seperti contoh berikut:

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

> **Note**
>
> Nilai-nilai untuk prinsipal dalam kebijakan kepercayaan untuk peran tersebut spesifik untuk IDP. Peran untuk hanya OIDC dapat menentukan satu prinsipal. Oleh karena itu, jika aplikasi seluler memungkinkan pengguna untuk masuk dari lebih dari satu iDP, buat peran terpisah untuk setiap IDP yang ingin Anda dukung. Buat kebijakan kepercayaan terpisah untuk setiap IDP.

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Login with Amazon, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh, *amzn1.application-oa2-123456* mewakili ID aplikasi yang ditetapkan Amazon saat Anda mengonfigurasi aplikasi menggunakan Login with Amazon.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
```

```
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Facebook, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh ini, *111222333444555* mewakili ID aplikasi yang ditetapkan Facebook.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Google, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh ini, *666777888999000* mewakili ID aplikasi yang ditetapkan Google.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Amazon Cognito, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh ini, *us-east:12345678-ffff-ffff-ffff-123456* mewakili ID kumpulan identitas yang ditetapkan Amazon Cognito.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

Menciptakan peran untuk OIDC

Setelah Anda menyelesaikan prasyarat, Anda dapat membuat peran di. IAM Prosedur berikut menjelaskan cara membuat peran OIDC federasi di AWS Management Console. Untuk membuat peran dari AWS CLI atau AWS API, lihat prosedur diBuat peran untuk penyedia identitas pihak ketiga (federasi).

> **Important**
>
> Jika Anda menggunakan Amazon Cognito, gunakan konsol Amazon Cognito untuk mengatur peran. Jika tidak, gunakan IAM konsol untuk membuat peran OIDC federasi.

Untuk membuat IAM peran untuk OIDC federasi

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, silakan pilih Peran lalu pilih Buat peran.
3. Pilih identitas Web sebagai jenis entitas tepercaya dan pilih Berikutnya.
4. Untuk penyedia Identity, pilih iDP untuk peran Anda:

- Jika Anda ingin membuat peran untuk IDP web individual, pilih Login with Amazon, Facebook, atau Google.

> **Note**
>
> Anda harus membuat peran terpisah untuk setiap IDP yang ingin Anda dukung.

- Jika Anda ingin membuat peran skenario lanjutan untuk Amazon Cognito, pilih Amazon Cognito.

> **Note**
>
> Anda harus membuat peran secara manual untuk digunakan dengan Amazon Cognito hanya ketika Anda mengerjakan skenario lanjutan. Jika tidak, Amazon Cognito dapat membuat peran untuk Anda. Untuk informasi selengkapnya tentang Amazon Cognito, lihat Penyedia identitas eksternal kumpulan identitas (identitas gabungan) di Panduan Pengembang Amazon Cognito.

- Jika Anda ingin membuat peran untuk GitHub Tindakan, Anda harus mulai dengan menambahkan GitHub OIDC penyedia keIAM. Setelah Anda menambahkan GitHub OIDC penyedia, pilih IAM token.actions.githubusercontent.com.

> **Note**
>
> Untuk informasi tentang cara AWS mengonfigurasi trust OIDC sebagai GitHub identitas federasi, lihat GitHub Docs - Mengonfigurasi OpenID Connect di Amazon Web Services. Untuk informasi tentang praktik terbaik untuk membatasi akses peran yang terkait dengan IAM IDP GitHub, lihat Mengkonfigurasi peran untuk penyedia GitHub OIDC identitas di halaman ini.

5. Masukkan pengenal untuk aplikasi Anda. Label pengenal berubah berdasarkan penyedia yang Anda pilih:

   - Jika Anda ingin membuat peran untuk Login with Amazon, masukkan ID aplikasi ke dalam kotak ID Aplikasi.
   - Jika Anda ingin membuat peran untuk Facebook, masukkan ID aplikasi ke dalam kotak ID Aplikasi.

- Jika Anda ingin membuat peran untuk Google, masukkan nama audiens ke dalam kotak Audiens.
- Jika Anda ingin membuat peran untuk Amazon Cognito, masukkan ID kumpulan identitas yang telah Anda buat untuk aplikasi Amazon Cognito ke dalam kotak ID Kumpulan Identitas.
- Jika Anda ingin membuat peran untuk GitHub Tindakan, masukkan detail berikut:
  - Untuk Audiens, pilih `sts.amazonaws.com`.
  - Untuk GitHub organisasi, masukkan nama GitHub organisasi Anda. Nama GitHub organisasi diperlukan dan harus alfanumerik termasuk tanda hubung (-). Anda tidak dapat menggunakan karakter wildcard (* dan?) dalam nama GitHub organisasi.
  - (Opsional) Untuk GitHub repositori, masukkan nama GitHub repositori. Jika Anda tidak menentukan nilai, itu default ke wildcard (). *
  - (Opsional) Untuk GitHub cabang, masukkan nama GitHub cabang. Jika Anda tidak menentukan nilai, itu default ke wildcard (). *

6. (Opsional) Untuk Kondisi (opsional), pilih Tambahkan Kondisi untuk membuat kondisi tambahan yang harus dipenuhi sebelum pengguna aplikasi Anda dapat menggunakan izin yang diberikan peran. Misalnya, Anda dapat menambahkan kondisi yang memberikan akses ke AWS sumber daya hanya untuk ID IAM pengguna tertentu. Anda juga dapat menambahkan ketentuan ke kebijakan kepercayaan setelah peran dibuat. Untuk informasi selengkapnya, lihat Memperbarui kebijakan kepercayaan peran .
7. Tinjau OIDC informasi Anda dan kemudian pilih Berikutnya.
8. IAMmenyertakan daftar kebijakan AWS terkelola dan terkelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk kebijakan izin, atau pilih Buat kebijakan untuk membuka tab browser baru dan membuat kebijakan baru dari awal. Untuk informasi selengkapnya, lihat Membuat IAM kebijakan. Setelah Anda membuat kebijakan, tutup tab tersebut dan kembali ke tab asli Anda. Pilih kotak centang di samping kebijakan izin yang ingin dimiliki OIDC pengguna. Jika Anda lebih suka, Anda boleh tidak memilih kebijakan saat ini, kemudian melampirkan kebijakan kepada peran di lain waktu. Secara default, peran tidak memiliki izin.
9. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan.

   Buka bagian batas izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. Pilih kebijakan yang akan digunakan untuk batas izin.
10. Pilih Berikutnya.
11. Untuk Nama peran, masukkan nama peran. Nama peran harus unik di dalam diri Anda Akun AWS. Mereka tidak bergantung pada kasus. Misalnya, Anda tidak dapat membuat peran

bernama keduanya **`PRODROLE`** dan**`prodrole`**. Karena AWS sumber daya lain mungkin mereferensikan peran, Anda tidak dapat mengedit nama peran setelah membuatnya.

12. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
13. Untuk mengedit kasus penggunaan dan izin untuk peran, pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin bagian.
14. (Opsional) Untuk menambahkan metadata ke peran, lampirkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
15. Tinjau peran, lalu pilih Buat peran.

Mengkonfigurasi peran untuk penyedia GitHub OIDC identitas

Jika Anda menggunakan GitHub sebagai penyedia identitas OpenID Connect (OIDC) (IDP), praktik terbaik adalah membatasi entitas yang dapat mengambil peran yang terkait dengan IDP. IAM Bila Anda menyertakan pernyataan kondisi dalam kebijakan kepercayaan, Anda dapat membatasi peran ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan kunci kondisi `token.actions.githubusercontent.com:sub` dengan operator kondisi string untuk membatasi akses. Kami menyarankan Anda membatasi kondisi ke kumpulan repositori atau cabang tertentu dalam organisasi Anda GitHub . Untuk informasi tentang cara AWS mengonfigurasi trust OIDC sebagai GitHub identitas federasi, lihat GitHub Docs - Mengonfigurasi OpenID Connect di Amazon Web Services.

Jika Anda menggunakan GitHub lingkungan dalam alur kerja tindakan atau OIDC kebijakan, kami sangat menyarankan untuk menambahkan aturan perlindungan ke lingkungan untuk keamanan tambahan. Gunakan cabang dan tag penyebaran untuk membatasi cabang dan tag mana yang dapat diterapkan ke lingkungan. Untuk informasi selengkapnya tentang mengonfigurasi lingkungan dengan aturan perlindungan, lihat Cabang dan tag penerapan di GitHub artikel Menggunakan lingkungan untuk penerapan.

Kapan GitHub OIDC idP adalah Principal tepercaya untuk peran Anda, IAM periksa kondisi kebijakan kepercayaan peran untuk memverifikasi bahwa kunci kondisi `token.actions.githubusercontent.com:sub` ada dan nilainya bukan semata-mata karakter wildcard (* dan?) atau null. IAMmelakukan pemeriksaan ini ketika kebijakan kepercayaan dibuat atau diperbarui. Jika kunci kondisi tidak `token.actions.githubusercontent.com:sub` ada, atau nilai kunci tidak memenuhi kriteria nilai yang disebutkan, permintaan akan gagal dan mengembalikan kesalahan.

> **Important**
>
> Jika Anda tidak membatasi kunci kondisi `token.actions.githubusercontent.com:sub` untuk organisasi atau repositori tertentu, maka GitHub Tindakan dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan iDP GitHub IAM di akun Anda. AWS

Contoh kebijakan kepercayaan berikut membatasi akses ke GitHub organisasi, repositori, dan cabang yang ditentukan. `token.actions.githubusercontent.com:sub`Nilai kunci kondisi dalam contoh berikut adalah format nilai subjek default yang didokumentasikan oleh GitHub.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

Contoh kondisi berikut membatasi akses ke GitHub organisasi dan repositori yang ditentukan, tetapi memberikan akses ke cabang mana pun dalam repositori.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
```

```
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}
```

Contoh kondisi berikut membatasi akses ke repositori atau cabang apa pun dalam organisasi yang ditentukan GitHub . Kami menyarankan Anda membatasi kunci kondisi `token.actions.githubusercontent.com:sub` ke nilai tertentu yang membatasi akses ke GitHub Tindakan dari dalam GitHub organisasi Anda.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
  }
}
```

Untuk informasi selengkapnya tentang kunci OIDC federasi yang tersedia untuk pemeriksaan kondisi dalam kebijakan, lihatKunci yang tersedia untuk AWS OIDC federasi.

Buat peran untuk federasi SAML 2.0 (konsol)

Anda dapat menggunakan federasi SAML 2.0 alih-alih membuat IAM pengguna di Akun AWS. Dengan penyedia identitas (iDP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk mengakses AWS sumber daya di akun Anda. Untuk informasi lebih lanjut tentang federasi dan penyedia identitas, lihat Penyedia dan federasi identitas.

> **Note**
>
> Untuk meningkatkan ketahanan federasi, kami menyarankan Anda mengonfigurasi IDP dan AWS federasi Anda untuk mendukung beberapa titik akhir masuk. SAML Untuk detailnya, lihat artikel Blog AWS Keamanan Cara menggunakan SAML titik akhir regional untuk failover.

Prasyarat untuk menciptakan peran untuk SAML

Sebelum Anda dapat membuat peran untuk federasi SAML 2.0, Anda harus terlebih dahulu menyelesaikan langkah-langkah prasyarat berikut.

Untuk mempersiapkan diri untuk membuat peran untuk federasi SAML 2.0

1. Sebelum Anda membuat peran untuk federasi SAML berbasis, Anda harus membuat SAML penyedia diIAM. Untuk informasi selengkapnya, lihat Buat penyedia SAML identitas di IAM.
2. Siapkan kebijakan untuk peran yang akan diasumsikan oleh pengguna yang SAML diautentikasi 2.0. Seperti halnya peran apa pun, peran SAML federasi mencakup dua kebijakan. Salah satunya adalah kebijakan kepercayaan peran yang dapat mengasumsikan peran. Yang lainnya adalah kebijakan IAM izin yang menentukan AWS tindakan dan sumber daya yang diizinkan atau ditolak aksesnya oleh pengguna federasi.

   Ketika Anda membuat kebijakan kepercayaan untuk peran Anda, Anda harus menggunakan tiga nilai untuk memastikan bahwa hanya aplikasi Anda yang dapat mengambil peran:

   - Untuk elemen `Action`, gunakan tindakan `sts:AssumeRoleWithSAML`.
   - Untuk elemen `Principal`, gunakan string `{"Federated":`*`ARNofIdentityProvider`*`}`. Ganti *`ARNofIdentityProvider`* dengan penyedia SAML identitas yang Anda buatStep 1. ARN
   - Untuk `Condition` elemen, gunakan `StringEquals` kondisi untuk menguji apakah `saml:aud` atribut dari SAML respons cocok dengan titik akhir SAML federasi. AWS

   Contoh kebijakan kepercayaan berikut dirancang untuk pengguna SAML federasi:

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
         "Effect": "Allow",
         "Action": "sts:AssumeRoleWithSAML",
         "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/PROVIDER-
   NAME"},
         "Condition": {"StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/
   saml"}}
       }
     }
   ```

   Ganti prinsipal ARN dengan yang sebenarnya ARN untuk SAML penyedia yang Anda buatIAM. Itu akan memiliki ID akun dan nama penyedia Anda sendiri.

Menciptakan peran untuk SAML

Setelah Anda menyelesaikan langkah-langkah prasyarat, Anda dapat membuat peran untuk SAML federasi berbasis.

Untuk membuat peran untuk federasi SAML berbasis

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi IAM konsol, pilih Peran, lalu pilih Buat peran.
3. Pilih jenis peran federasi SAML 2.0.
4. Untuk Pilih SAML penyedia, pilih penyedia untuk peran Anda.
5. Pilih metode tingkat akses SAML 2.0.

   - Pilih Izinkan akses terprogram hanya untuk membuat peran yang dapat diasumsikan secara terprogram dari atau. AWS API AWS CLI
   - Pilih Izinkan programatik dan AWS Management Console akses untuk membuat peran yang dapat diasumsikan secara terprogram dan dari. AWS Management Console

   Peran yang dibuat oleh keduanya serupa, tetapi peran yang juga dapat diasumsikan dari konsol mencakup suatu kebijakan kepercayaan dengan syarat tertentu. Kondisi itu secara eksplisit memastikan bahwa SAML audiens (`SAML:aud`atribut) disetel ke titik akhir AWS masuk untuk SAML (https://signin.aws.amazon.com/saml).
6. Jika Anda membuat peran untuk akses terprogram, pilih atribut dari daftar Atribut. Kemudian, di kotak Nilai, masukkan nilai untuk disertakan dalam peran. Ini membatasi akses peran ke pengguna dari penyedia identitas yang respons SAML autentikasi (pernyataan) menyertakan atribut yang Anda tentukan. Anda harus menentukan setidaknya satu atribut untuk memastikan bahwa peran Anda terbatas pada subset pengguna dalam organisasi Anda.

   Jika Anda membuat peran untuk akses terprogram dan konsol, `SAML:aud` atribut secara otomatis ditambahkan dan disetel ke AWS SAML titik URL akhir (https://signin.aws.amazon.com/saml).
7. Untuk menambahkan lebih banyak kondisi terkait atribut ke kebijakan trust, pilih Kondisi (opsional), pilih kondisi tambahan, dan tentukan nilai.

> **Note**
>
> Daftar ini mencakup SAML atribut yang paling umum digunakan. IAMmendukung atribut tambahan yang dapat Anda gunakan untuk membuat kondisi. Untuk daftar atribut yang didukung, lihat Kunci yang Tersedia untuk SAML Federasi. Jika Anda memerlukan kondisi untuk SAML atribut yang didukung yang tidak ada dalam daftar, Anda dapat menambahkan kondisi tersebut secara manual. Untuk melakukannya, ubah kebijakan kepercayaan setelah Anda membuat peran tersebut.

8. Tinjau informasi kepercayaan SAML 2.0 Anda dan kemudian pilih Berikutnya.
9. IAMmenyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk kebijakan izin, atau pilih Buat kebijakan untuk membuka tab browser baru dan membuat kebijakan baru dari awal. Untuk informasi selengkapnya, lihat Membuat IAM kebijakan. Setelah Anda membuat kebijakan, tutup tab tersebut dan kembali ke tab asli Anda. Pilih kotak centang di samping kebijakan izin yang ingin dimiliki OIDC oleh pengguna federasi. Jika Anda lebih suka, Anda boleh tidak memilih kebijakan saat ini, kemudian melampirkan kebijakan kepada peran di lain waktu. Secara default, peran tidak memiliki izin.
10. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan.

    Buka bagian batas izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. Pilih kebijakan yang akan digunakan untuk batas izin.
11. Pilih Berikutnya.
12. Pilih Berikutnya: Tinjau.
13. Untuk Nama peran, masukkan nama peran. Nama peran harus unik dalam diri Anda Akun AWS. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat peran dengan nama **`PRODROLE`** dan **`prodrole`**. Karena AWS sumber daya lain mungkin merujuk peran, Anda tidak dapat mengedit nama peran setelah dibuat.
14. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
15. Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan bagian izin untuk mengedit kasus penggunaan dan izin untuk peran tersebut.
16. (Opsional) Tambahkan metadata ke dalam peran dengan melampirkan tanda sebagai pasangan nilai-kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
17. Tinjau peran, lalu pilih Buat peran.

Setelah Anda membuat peran, Anda menyelesaikan SAML kepercayaan dengan mengonfigurasi perangkat lunak penyedia identitas Anda dengan informasi tentang AWS. Informasi ini mencakup peran yang Anda inginkan agar digunakan oleh pengguna federasi Anda. Ini disebut sebagai mengonfigurasi kepercayaan pihak pengandal antara Idp dan AWS. Untuk informasi selengkapnya, lihat Konfigurasikan IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim.

Membuat peran menggunakan kebijakan kepercayaan khusus

Anda dapat membuat kebijakan kepercayaan khusus untuk mendelegasikan akses dan mengizinkan orang lain melakukan tindakan di situs Anda Akun AWS. Untuk informasi selengkapnya, lihat Membuat IAM kebijakan.

Untuk informasi tentang cara menggunakan peran untuk mendelegasikan izin, lihat Istilah dan konsep peran.

Membuat IAM peran menggunakan kebijakan kepercayaan khusus (konsol)

Anda dapat menggunakan AWS Management Console untuk membuat peran yang dapat diasumsikan IAM oleh pengguna. Misalnya, asumsikan bahwa organisasi Anda memiliki banyak Akun AWS untuk mengisolasi lingkungan pengembangan dari lingkungan produksi. Untuk informasi tingkat tinggi tentang membuat peran yang memungkinkan pengguna di akun pengembangan mengakses sumber daya di akun produksi, lihatContoh skenario menggunakan akun pengembangan dan produksi terpisah.

Untuk membuat peran menggunakan kebijakan kepercayaan khusus (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Pilih jenis peran kebijakan kepercayaan kustom.
4. Di bagian Kebijakan kepercayaan khusus, masukkan atau tempel kebijakan kepercayaan khusus untuk peran tersebut. Untuk informasi selengkapnya, lihat Membuat IAM kebijakan.
5. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.
6. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.

Buka bagian batas izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. IAMmenyertakan daftar kebijakan AWS terkelola dan terkelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk batas izin.

7. Pilih Berikutnya.
8. Untuk Nama peran, tingkat penyesuaian nama peran ditentukan oleh layanan. Jika layanan mendefinisikan nama peran, opsi ini tidak dapat diedit. Dalam kasus lain, layanan mungkin menentukan awalan peran dan memungkinkan Anda untuk memasukkan akhiran opsional. Beberapa layanan memungkinkan Anda untuk menentukan seluruh nama peran Anda.

   Jika memungkinkan, masukkan nama peran atau akhiran nama peran. Nama peran harus unik di dalam diri Anda Akun AWS. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat peran dengan nama **`PRODROLE`** dan **`prodrole`**. Karena AWS sumber daya lain mungkin mereferensikan peran, Anda tidak dapat mengedit nama peran setelah dibuat.
9. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
10. (Opsional) Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin bagian untuk mengedit kebijakan kustom dan izin untuk peran tersebut.
11. (Opsional) Tambahkan metadata ke dalam peran dengan melampirkan tanda sebagai pasangan nilai-kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
12. Tinjau peran dan kemudian pilih Buat peran.

Contoh kebijakan untuk mendelegasikan akses

Contoh berikut menunjukkan bagaimana Anda dapat mengizinkan atau memberikan Akun AWS akses ke sumber daya di tempat lain Akun AWS. Untuk mempelajari cara membuat IAM kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

Topik

- Menggunakan peran untuk mendelegasikan akses ke sumber daya orang lain Akun AWS sumber daya
- Menggunakan kebijakan untuk mendelegasikan akses ke layanan
- Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke bucket Amazon S3 di akun lain

- Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke antrian Amazon di akun lain SQS
- Tidak dapat mendelegasikan akun ketika akun ditolak aksesnya

Menggunakan peran untuk mendelegasikan akses ke sumber daya orang lain Akun AWS sumber daya

Untuk tutorial yang menunjukkan cara menggunakan IAM peran untuk memberikan akses kepada pengguna dalam satu akun AWS sumber daya yang ada di akun lain, lihatIAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran.

> **Important**
>
> Anda dapat menyertakan peran atau pengguna tertentu dalam `Principal` elemen kebijakan kepercayaan peran. ARN Saat Anda menyimpan kebijakan, AWS mengubah menjadi ARN ID utama yang unik. Hal ini membantu memitigasi risiko seseorang meningkatkan hak istimewa mereka dengan menghapus dan membuat kembali peran atau pengguna. Anda biasanya tidak melihat ID ini di konsol, karena ada juga transformasi terbalik kembali ke ARN saat kebijakan kepercayaan ditampilkan. Namun, jika Anda menghapus peran atau pengguna, maka hubungan Anda akan rusak. Kebijakan tidak lagi berlaku, bahkan jika Anda membuat ulang pengguna atau peran karena itu tidak sesuai dengan ID prinsipal yang disimpan dalam kebijakan kepercayaan. Ketika ini terjadi, ID utama muncul di konsol karena AWS tidak bisa lagi memetakannya kembali ke fileARN. Hasilnya adalah jika Anda menghapus dan membuat ulang pengguna atau peran yang direferensikan dalam `Principal` elemen kebijakan kepercayaan, Anda harus mengedit peran tersebut untuk mengganti peran tersebut. ARN Itu diubah menjadi ID prinsipal baru saat Anda menyimpan kebijakan.

Menggunakan kebijakan untuk mendelegasikan akses ke layanan

Contoh berikut ini menunjukkan kebijakan yang dapat dilampirkan pada sebuah peran. Kebijakan ini memungkinkan dua layanan, Amazon EMR dan AWS Data Pipeline untuk mengambil peran. Layanan kemudian dapat melakukan tugas yang diberikan oleh kebijakan izin yang ditetapkan untuk peran tersebut (tidak ditampilkan). Untuk menetapkan beberapa prinsipal layanan, Anda tidak menentukan dua elemen `Service`; Anda hanya dapat memiliki satu. Sebagai gantinya, Anda menggunakan serangkaian dari beberapa prinsipal layanan sebagai nilai elemen `Service` tunggal.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke bucket Amazon S3 di akun lain

Dalam contoh ini, akun A menggunakan kebijakan berbasis sumber daya (kebijakan bucket Amazon S3) untuk memberikan akun B akses penuh ke bucket S3 akun A. Kemudian akun B membuat kebijakan IAM pengguna untuk mendelegasikan akses ke bucket akun A ke salah satu pengguna di akun B.

Kebijakan S3 bucket di akun A mungkin terlihat seperti kebijakan berikut. Dalam contoh ini, bucket S3 akun A diberi nama amzn-s3-demo-bucket, dan nomor akun B adalah 111122223333. Itu tidak menentukan pengguna individual atau pengguna kelompok di akun B, hanya akun itu sendiri.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

Atau, akun A dapat menggunakan Amazon S3 Access Control Lists (ACLs) untuk memberikan akses akun B ke bucket S3 atau satu objek di dalam bucket. Dalam hal ini, satu-satunya hal yang berubah adalah bagaimana akun A memberikan akses ke akun B. Akun B masih menggunakan kebijakan untuk mendelegasikan akses ke IAM grup di akun B, seperti yang dijelaskan di bagian selanjutnya dari contoh ini. Untuk informasi selengkapnya tentang mengontrol akses pada bucket dan objek S3, buka Kontrol Akses di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Administrator akun B dapat membuat sampel kebijakan berikut. Kebijakan ini memungkinkan akses baca ke kelompok atau pengguna di akun B. Kebijakan sebelumnya memberikan akses ke akun B. Namun demikian, kelompok individu dan pengguna di akun B tidak dapat mengakses sumber daya sampai suatu kelompok atau pengguna memberikan izin secara jelas ke sumber daya tersebut. Izin dalam kebijakan ini hanya dapat menjadi subset dari izin yang ada di kebijakan lintas akun sebelumnya. Akun B tidak dapat memberikan lebih banyak izin untuk kelompok dan penggunanya daripada yang diberikan akun A ke akun B dalam kebijakan pertama. Dalam kebijakan ini, elemen `Action` secara jelas ditentukan untuk hanya mengizinkan tindakan `List`, dan elemen `Resource` kebijakan ini sesuai dengan `Resource` untuk kebijakan bucket yang diterapkan oleh akun A.

Untuk menerapkan kebijakan ini, akun B gunakan IAM untuk melampirkannya ke pengguna (atau grup) yang sesuai di akun B.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke antrian Amazon di akun lain SQS

Dalam contoh berikut, akun A memiliki SQS antrian Amazon yang menggunakan kebijakan berbasis sumber daya yang dilampirkan pada antrian untuk memberikan akses antrian ke akun B. Kemudian akun B menggunakan kebijakan IAM grup untuk mendelegasikan akses ke grup di akun B.

Contoh kebijakan antrean berikut memberi akun B izin untuk melakukan tindakan `SendMessage` dan `ReceiveMessage` pada antrean akun A yang disebut antrean1, tetapi hanya antara tengah hari hingga pukul 15.00 pada 30 November 2014. Nomor akun Akun B adalah 1111-2222-3333. Akun A menggunakan Amazon SQS untuk menerapkan kebijakan ini.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

Kebijakan akun B untuk mendelegasikan akses ke suatu kelompok di akun B dapat terlihat seperti contoh berikut. Akun B menggunakan IAM untuk melampirkan kebijakan ini ke grup (atau pengguna).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
  }
}
```

Dalam contoh kebijakan IAM pengguna sebelumnya, akun B menggunakan wildcard untuk memberikan akses penggunanya ke semua SQS tindakan Amazon pada antrean akun A. Namun, akun B hanya dapat mendelegasikan akses jika akun B telah diberi akses. Kelompok akun B yang memiliki kebijakan kedua dapat mengakses antrean hanya antara tengah hari hingga pukul 15.00 pada 30 November 2014. Pengguna hanya dapat melakukan `ReceiveMessage` tindakan `SendMessage` dan, sebagaimana didefinisikan dalam kebijakan SQS antrian Amazon akun A.

Tidak dapat mendelegasikan akun ketika akun ditolak aksesnya

Sesi Akun AWS tidak dapat mendelegasikan akses ke sumber daya akun lain jika akun lain secara eksplisit menolak akses ke akun induk pengguna. Penolakan ini meluas ke para pengguna dalam akun tersebut, baik apakah pengguna sudah memiliki kebijakan yang memberikan akses kepada mereka atau belum.

Sebagai contoh, akun A menyusun kebijakan bucket pada bucket S3 akun A yang secara eksplisit menolak akses akun B ke bucket akun A. Tetapi akun B menulis kebijakan IAM pengguna yang memberi pengguna di akun B akses ke keranjang akun A. Penolakan eksplisit yang diterapkan pada bucket S3 akun A menyebar ke pengguna di akun B. Ini mengesampingkan kebijakan IAM pengguna yang memberikan akses ke pengguna di akun B. (Untuk informasi terperinci bagaimana izin dievaluasi, lihat.) Logika evaluasi kebijakan

Kebijakan bucket Akun A dapat terlihat seperti kebijakan berikut ini. Dalam contoh ini, bucket S3 akun A diberi nama amzn-s3-demo-bucket, dan nomor akun B adalah 1111-2222-3333. Akun A menggunakan Amazon S3 untuk menerapkan kebijakan ini.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
  }
}
```

Penolakan secara eksplisit ini membatalkan kebijakan apa pun dalam akun B yang memberikan izin untuk mengakses bucket S3 dalam akun A.

IAMmanajemen peran

Sebelum pengguna, aplikasi, atau layanan dapat menggunakan peran yang Anda buat, Anda harus memberikan izin untuk beralih ke peran tersebut. Anda dapat menggunakan kebijakan apa pun yang dilampirkan ke grup atau pengguna untuk memberikan izin yang diperlukan. Bagian ini menjelaskan cara memberikan izin kepada pengguna untuk menggunakan peran. Ini juga menjelaskan bagaimana pengguna dapat beralih ke peran dari AWS Management Console, Alat untuk Windows PowerShell, AWS Command Line Interface (AWS CLI) dan `AssumeRole`API.

> **Important**
>
> Saat Anda membuat peran secara terprogram alih-alih di IAM konsol, Anda memiliki opsi untuk menambahkan hingga 512 karakter selain`RoleName`, yang panjangnya bisa mencapai 64 karakter. `Path` Namun, jika Anda bermaksud menggunakan peran dengan fitur Switch Role di AWS Management Console, maka gabungan `Path` dan `RoleName` tidak dapat melebihi 64 karakter.

Topik

Lihat akses peran

Sebelum mengubah izin pengguna, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya tentang melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghasilkan kebijakan berdasarkan informasi akses

Terkadang Anda dapat memberikan izin kepada IAM entitas (pengguna atau peran) di luar yang mereka butuhkan. Untuk membantu Anda menyaring izin yang Anda berikan, Anda dapat membuat IAM kebijakan yang didasarkan pada aktivitas akses untuk entitas. IAMAccess Analyzer meninjau AWS CloudTrail log Anda dan membuat templat kebijakan yang berisi izin yang telah digunakan oleh entitas dalam rentang tanggal yang ditentukan. Anda dapat menggunakan templat untuk membuat

kebijakan terkelola dengan izin berbutir halus, lalu melampirkannya ke entitas. IAM Dengan begitu, Anda hanya memberikan izin yang dibutuhkan pengguna atau peran untuk berinteraksi dengan AWS sumber daya untuk kasus penggunaan spesifik Anda. Untuk mempelajari selengkapnya, lihat Pembuatan kebijakan IAM Access Analyzer.

Berikan izin pengguna untuk beralih peran

Ketika administrator membuat peran untuk akses lintas akun, mereka membangun kepercayaan antara akun yang memiliki peran, sumber daya (akun kepercayaan), dan akun yang berisi pengguna (akun tepercaya). Untuk melakukannya, administrator akun terpercaya menentukan nomor akun terpercaya sebagai `Principal` dalam kebijakan kepercayaan peran tersebut. Itu berpotensi memungkinkan pengguna mana pun di akun tepercaya untuk mengambil peran tersebut. Untuk menyelesaikan konfigurasi, administrator akun terpercaya harus memberikan izin kepada kelompok atau pengguna tertentu dalam izin akun tersebut untuk beralih ke peran.

Untuk memberikan izin untuk beralih ke peran

1. Sebagai administrator akun tepercaya, buat kebijakan baru untuk pengguna, atau edit kebijakan yang ada untuk menambahkan elemen yang diperlukan. Untuk detailnya, lihat Membuat atau mengubah.
2. Kemudian, pilih cara Anda ingin membagikan informasi peran:

   - Tautan peran: Kirim pengguna tautan yang membawa mereka ke halaman Beralih Peran dengan semua detail yang sudah diisi.
   - ID Akun atau alias: Berikan setiap pengguna nama peran bersama dengan nomor ID akun atau alias akun. Pengguna kemudian masuk ke halaman Ganti Peran dan menambahkan detail secara manual.

   Untuk detailnya, lihat Memberikan informasi kepada pengguna.

Perhatikan bahwa Anda dapat beralih peran hanya jika Anda masuk sebagai IAM pengguna, peran SAML -federasi, atau peran federasi identitas web. Anda tidak dapat beralih peran saat masuk sebagai Pengguna root akun AWS.

Important

Anda tidak dapat beralih peran AWS Management Console ke peran yang membutuhkan ExternalIdnilai. Anda dapat beralih ke peran seperti itu hanya dengan memanggil `AssumeRole`APIyang mendukung `ExternalId` parameter.

Catatan

- Topik ini membahas kebijakan untuk pengguna, karena pada akhirnya Anda memberikan izin kepada pengguna untuk menyelesaikan tugas. Namun, kami tidak menyarankan Anda memberikan izin langsung ke pengguna individu. Saat pengguna mengambil peran, mereka diberi izin yang terkait dengan peran tersebut.
- Saat Anda beralih peran AWS Management Console, konsol selalu menggunakan kredenal asli Anda untuk mengotorisasi sakelar. Ini berlaku baik Anda masuk sebagai IAM pengguna, sebagai peran SAML -federasi, atau sebagai peran federasi identitas web. Misalnya, jika Anda beralih ke RoleAIAM, gunakan pengguna asli atau kredenal peran federasi untuk menentukan apakah Anda diizinkan untuk mengambil RoleA. Jika Anda kemudian mencoba untuk beralih ke RoleB saat menggunakan RoleA, pengguna asli atau kredensial peran federasi digunakan untuk mengotorisasi upaya Anda. Kredensial untuk RoleA tidak digunakan untuk tindakan ini.

Topik

Membuat atau mengubah

Kebijakan yang memberikan izin kepada pengguna untuk mengasumsikan peran harus mencantumkan pernyataan dengan efek `Allow` pada hal berikut:

- Tindakan `sts:AssumeRole`
- Amazon Resource Name (ARN) dari peran dalam `Resource` elemen

Pengguna yang mendapatkan kebijakan diizinkan untuk beralih peran pada sumber daya yang terdaftar (baik melalui keanggotaan grup atau langsung dilampirkan).

> **Note**
>
> Jika `Resource` diatur menjadi `*`, pengguna dapat mengasumsikan peran apa pun dalam akun apa pun yang memercayai akun pengguna. (Dengan kata lain, kebijakan kepercayaan peran menentukan akun pengguna sebagai `Principal`). Sebagai praktik terbaik, kami menyarankan Anda mengikuti prinsip hak istimewa paling sedikit dan menentukan yang lengkap hanya ARN untuk peran yang dibutuhkan pengguna.

Contoh berikut menunjukkan kebijakan yang memungkinkan pengguna untuk mengasumsikan peran hanya dalam satu akun. Selain itu, kebijakan ini menggunakan wildcard (*) untuk menentukan bahwa pengguna dapat beralih ke peran hanya jika nama peran dimulai dan huruf `Test`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account-id:role/Test*"
  }
}
```

> **Note**
>
> Izin yang diberikan oleh peran kepada pengguna tidak menambah izin yang sudah diberikan kepada pengguna. Saat pengguna beralih ke suatu peran, pengguna sementara waktu akan memberikan izin awalnya sebagai ganti atas apa yang diberikan oleh peran tersebut. Saat pengguna keluar dari peran, izin pengguna asli mereka dipulihkan secara otomatis. Misalnya, misalkan izin pengguna mengizinkan bekerja dengan EC2 instans Amazon, tetapi kebijakan izin peran tidak memberikan izin tersebut. Dalam hal ini, saat menggunakan peran, pengguna tidak dapat bekerja dengan EC2 instance Amazon di konsol. Selain itu, kredensi sementara yang diperoleh melalui `AssumeRole` tidak berfungsi dengan EC2 instans Amazon secara terprogram.

Memberikan informasi kepada pengguna

Setelah Anda membuat peran dan memberikan izin kepada pengguna Anda untuk beralih ke sana, Anda harus memberikan hal-hal berikut kepada pengguna:

- Nama peran
- ID atau alias akun yang berisi peran tersebut

Anda dapat merampingkan akses untuk pengguna Anda dengan mengirimkan tautan yang telah dikonfigurasi sebelumnya dengan ID akun dan nama peran. Anda dapat melihat tautan peran setelah menyelesaikan wizard Buat Peran dengan memilih spanduk Lihat Peran, atau pada halaman Ringkasan Peran untuk peran yang diaktifkan lintas akun.

Anda juga dapat menggunakan format berikut untuk menyusun tautan secara manual. Ganti ID akun atau alias Anda dan nama peran untuk dua parameter dalam contoh berikut.

`https://signin.aws.amazon.com/switchrole?account=`*`your_account_ID_or_alias`*`&roleName=`*`optional_path/role_name`*

Kami menyarankan Anda mengarahkan pengguna Anda Beralih dari pengguna ke IAM peran (konsol) untuk memandu mereka melalui proses tersebut. Untuk memecahkan masalah umum yang mungkin Anda temui saat mengasumsikan peran, lihat Saya tidak dapat mengsumsikan peran.

Pertimbangan

- Jika Anda membuat peran secara terprogram, Anda dapat membuat peran dengan jalur dan nama. Jika demikian, Anda harus memberikan jalur lengkap dan nama peran kepada pengguna agar mereka dapat masuk ke halaman Beralih Peran di AWS Management Console. Sebagai contoh: `division_abc/subdivision_efg/role_XYZ`.
- Jika Anda membuat peran secara terprogram, Anda dapat menambahkan hingga 512 karakter dan a`Path`. `RoleName` Panjang nama peran dapat mencapai 64 karakter. Namun, untuk menggunakan peran dengan fitur Switch Role di AWS Management Console, gabungan `Path` dan `RoleName` tidak dapat melebihi 64 karakter.

- Untuk tujuan keamanan, Anda dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan AWS. Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan identias saat mereka mengasumsikan sebuah peran. Misalnya, Anda dapat meminta IAM pengguna

menentukan nama pengguna mereka sendiri sebagai identitas sumber mereka. Ini dapat membantu Anda menentukan pengguna mana yang melakukan tindakan tertentu di AWS. Untuk informasi selengkapnya, lihat `sts:SourceIdentity`. Anda juga dapat menggunakan `sts:RoleSessionName` untuk mengharuskan pengguna menentukan nama sesi saat mereka mengambil peran. Hal ini dapat membantu Anda membedakan antara sesi peran ketika peran digunakan oleh prinsipal yang berbeda.

Berikan izin pengguna untuk meneruskan peran ke layanan AWS

Untuk mengkonfigurasi banyak AWS layanan, Anda harus memberikan IAM peran ke layanan. Ini memungkinkan layanan untuk mengambil peran nanti dan melakukan tindakan atas nama Anda. Untuk sebagian besar layanan, Anda hanya perlu meneruskan peran ke layanan satu kali selama penyiapan, dan tidak setiap kali layanan mengambil peran tersebut. Misalnya, asumsikan bahwa Anda memiliki aplikasi yang berjalan pada EC2 instance Amazon. Aplikasi tersebut memerlukan kredensial sementara untuk autentikasi, dan izin untuk mengotorisasi aplikasi guna melakukan tindakan di AWS. Saat menyiapkan aplikasi, Anda harus meneruskan peran ke Amazon EC2 untuk digunakan dengan instance yang menyediakan kredensi tersebut. Anda menentukan izin untuk aplikasi yang berjalan pada instance dengan melampirkan IAM kebijakan ke peran. Aplikasi ini mengasumsikan peran setiap kali diperlukan untuk melakukan tindakan yang diizinkan oleh peran tersebut.

Untuk meneruskan peran (dan izinnya) ke AWS layanan, pengguna harus memiliki izin untuk meneruskan peran tersebut ke layanan. Ini membantu administrator untuk memastikan bahwa hanya pengguna yang disetujui yang dapat mengonfigurasi layanan dengan peran yang memberikan izin. Untuk mengizinkan pengguna meneruskan peran ke AWS layanan, Anda harus memberikan `PassRole` izin kepada IAM pengguna, peran, atau grup pengguna.

> **Warning**
>
> - Anda hanya dapat menggunakan `PassRole` izin untuk meneruskan IAM peran ke layanan yang berbagi AWS akun yang sama. Untuk meneruskan peran di Akun A ke layanan di Akun B, Anda harus terlebih dahulu membuat IAM peran di Akun B yang dapat mengambil peran dari Akun A, dan kemudian peran dalam Akun B dapat diteruskan ke layanan. Untuk detailnya, lihat Akses sumber daya lintas akun di IAM.
> - Jangan mencoba mengontrol siapa yang dapat melewati peran dengan menandai peran dan kemudian menggunakan kunci `ResourceTag` kondisi dalam kebijakan dengan

`iam:PassRole` tindakan tersebut. Pendekatan ini tidak memiliki hasil yang dapat diandalkan.

Saat menyetel `PassRole` izin, Anda harus memastikan bahwa pengguna tidak melewati peran di mana peran tersebut memiliki lebih banyak izin daripada yang Anda inginkan untuk dimiliki pengguna. Misalnya, Alice mungkin tidak diizinkan untuk melakukan tindakan Amazon S3 apa pun. Jika Alice dapat meneruskan peran ke layanan yang memungkinkan tindakan Amazon S3, layanan dapat melakukan tindakan Amazon S3 atas nama Alice saat menjalankan pekerjaan.

Saat menentukan peran terkait layanan, Anda juga harus memiliki izin untuk meneruskan peran tersebut ke layanan. Beberapa layanan secara otomatis membuat peran yang terkait dengan layanan di akun Anda ketika Anda melakukan tindakan di layanan tersebut. Misalnya, Amazon EC2 Auto Scaling membuat peran `AWSServiceRoleForAutoScaling` terkait layanan untuk Anda saat Anda membuat grup Auto Scaling untuk pertama kalinya. Jika Anda mencoba menentukan peran terkait layanan saat membuat grup Auto Scaling dan Anda tidak memiliki `iam:PassRole` izin, Anda akan menerima kesalahan. Jika Anda tidak menentukan peran secara eksplisit, `iam:PassRole` izin tidak diperlukan, dan defaultnya adalah menggunakan `AWSServiceRoleForAutoScaling` peran untuk semua operasi yang dilakukan pada grup tersebut. Untuk mempelajari layanan yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari layanan mana yang secara otomatis membuat peran yang terkait dengan layanan saat Anda melakukan tindakan dalam layanan tersebut, pilih tautan Ya dan lihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Pengguna dapat meneruskan peran ARN sebagai parameter dalam API operasi apa pun yang menggunakan peran tersebut untuk menetapkan izin ke layanan. Layanan kemudian memeriksa apakah pengguna memiliki izin `iam:PassRole`. Untuk membatasi pengguna agar hanya meneruskan peran yang disetujui, Anda dapat memfilter `iam:PassRole` izin dengan `Resources` elemen pernyataan IAM kebijakan.

Anda dapat menggunakan `Condition` elemen dalam JSON kebijakan untuk menguji nilai kunci yang disertakan dalam konteks permintaan semua AWS permintaan. Untuk mempelajari penggunaan kunci syarat dalam kebijakan, lihat IAMJSONelemen kebijakan: Condition. Kunci syarat `iam:PassedToService` dapat digunakan untuk menentukan ke prinsipal layanan mana sebuah peran dapat diberikan. Untuk mempelajari selengkapnya tentang menggunakan kunci `iam:PassedToService` kondisi dalam kebijakan, lihat iam: PassedToService.

Contoh 1

Misalkan Anda ingin memberi pengguna kemampuan untuk meneruskan serangkaian peran yang disetujui ke EC2 layanan Amazon setelah meluncurkan instance. Anda memerlukan tiga elemen:

- Kebijakan IAM izin yang dilampirkan pada peran yang menentukan apa yang dapat dilakukan peran tersebut. Cakupan diperbolehkan hanya untuk tindakan yang harus dilakukan peran, dan hanya untuk sumber daya yang diperlukan peran untuk tindakan tersebut. Anda dapat menggunakan kebijakan IAM izin AWS terkelola atau dibuat pelanggan.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [ "A list of the permissions the role is allowed to use" ],
        "Resource": [ "A list of the resources the role is allowed to access" ]
    }
}
```

- Kebijakan kepercayaan untuk peran yang memungkinkan layanan untuk mengasumsikan peran. Misalnya, Anda dapat melampirkan kebijakan kepercayaan berikut pada peran dengan tindakan `UpdateAssumeRolePolicy`. Kebijakan kepercayaan ini memungkinkan Amazon EC2 untuk menggunakan peran dan izin yang dilampirkan pada peran tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
        "Effect": "Allow",
        "Principal": { "Service": "ec2.amazonaws.com" },
       "Action": "sts:AssumeRole"
    }
}
```

- Kebijakan IAM izin yang dilampirkan pada IAM pengguna yang memungkinkan pengguna untuk hanya meneruskan peran yang disetujui tersebut. Anda biasanya `iam:GetRole` menambahkan `iam:PassRole` sehingga pengguna bisa mendapatkan detail peran yang akan diteruskan. Dalam contoh ini, pengguna hanya dapat meneruskan peran yang ada di akun yang ditentukan dengan nama yang diawali dengan`EC2-roles-for-XYZ-`:

```
{
    "Version": "2012-10-17",
    "Statement": [{
```

```
        "Effect": "Allow",
        "Action": [
            "iam:GetRole",
            "iam:PassRole"
        ],
        "Resource": "arn:aws:iam::account-id:role/EC2-roles-for-XYZ-*"
    }]
}
```

Sekarang pengguna dapat memulai EC2 instance Amazon dengan peran yang ditetapkan. Aplikasi yang berjalan pada instans dapat mengakses kredensial sementara untuk peran tersebut melalui metadata profil instans. Kebijakan izin IAM yang dilamprkan ke peran yang menentukan apa yang dapat dilakukan oleh instans tersebut.

Contoh 2

Amazon Relational Database Service (RDSAmazon) mendukung fitur yang disebut Enhanced Monitoring. Fitur ini memungkinkan Amazon RDS untuk memantau instance database menggunakan agen. Ini juga memungkinkan Amazon RDS untuk mencatat metrik ke Amazon CloudWatch Logs. Untuk mengaktifkan fitur ini, Anda harus membuat peran layanan untuk memberikan RDS izin Amazon untuk memantau dan menulis metrik ke log Anda.

Untuk membuat peran bagi pemantauan Amazon yang RDS disempurnakan

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Peran, lalu pilih Buat peran.
3. Pilih jenis peran AWS Layanan, lalu untuk kasus Penggunaan untuk lainnya Layanan AWS, pilih RDSlayanan. Pilih RDS— Pemantauan yang Ditingkatkan, lalu pilih Berikutnya.
4. Pilih kebijakan mazonRDSEnhanced MonitoringRole izin A.
5. Pilih Berikutnya.
6. Untuk nama Peran, masukkan nama peran yang membantu Anda mengidentifikasi tujuan peran ini. Nama peran harus unik dalam diri Anda Akun AWS. Ketika nama peran digunakan dalam kebijakan atau sebagai bagian dari sebuahARN, nama peran tersebut peka huruf besar/kecil. Saat nama peran muncul ke pelanggan di konsol, seperti selama proses masuk, nama peran tidak peka huruf besar/kecil. Karena berbagai entitas mungkin mereferensikan peran, Anda tidak dapat mengedit nama peran setelah dibuat.

7. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
8. (Opsional) Tambahkan metadata ke pengguna dengan cara melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
9. Tinjau peran dan kemudian pilih Buat peran.

Peran tersebut secara otomatis mendapatkan kebijakan kepercayaan yang memberikan izin layanan `monitoring.rds.amazonaws.com` untuk memegang peran tersebut. Setelah itu, Amazon RDS dapat melakukan semua tindakan yang diizinkan oleh `AmazonRDSEnhancedMonitoringRole` kebijakan tersebut.

Pengguna yang ingin Anda akses Enhanced Monitoring memerlukan kebijakan yang menyertakan pernyataan yang memungkinkan pengguna untuk membuat daftar RDS peran dan pernyataan yang memungkinkan pengguna untuk meneruskan peran, seperti berikut ini. Gunakan nomor akun Anda dan ganti nama peran dengan nama yang Anda berikan di langkah 6.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
    "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
    "Effect": "Allow",
    "Action": [ "iam:PassRole" ],
    "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
}
```

Anda dapat menggabungkan pernyataan ini dengan pernyataan dalam kebijakan lain atau menempatkannya ke dalam kebijakannya sendiri. Sebagai gantinya, untuk menentukan bahwa pengguna dapat meneruskan peran apa pun yang dimulaiRDS-, Anda dapat mengganti nama peran di sumber daya ARN dengan wildcard, sebagai berikut.

```
    "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

`iam:PassRole`tindakan dalam AWS CloudTrail log

`PassRole`Bukan API panggilan. `PassRole`adalah izin, artinya tidak ada CloudTrail log yang dihasilkan untuk IAM`PassRole`. Untuk meninjau peran apa yang diteruskan ke mana Layanan AWS CloudTrail, Anda harus meninjau CloudTrail log yang membuat atau memodifikasi AWS sumber daya yang menerima peran tersebut. Misalnya, peran diteruskan ke AWS Lambda fungsi saat dibuat. Log untuk `CreateFunction` tindakan menunjukkan catatan peran yang diteruskan ke fungsi.

Mencabut kredensi IAM keamanan sementara peran

> **Warning**
>
> Jika Anda mengikuti langkah-langkah di halaman ini, semua pengguna dengan sesi saat ini yang dibuat dengan asumsi peran ditolak akses ke semua AWS tindakan dan sumber daya. Hal ini dapat mengakibatkan pengguna kehilangan pekerjaan yang belum disimpan.

Ketika Anda mengizinkan pengguna untuk mengakses AWS Management Console dengan waktu durasi sesi yang lama (seperti 12 jam), kredensi sementara mereka tidak kedaluwarsa dengan cepat. Jika pengguna secara tidak sengaja mengekspos kredensialnya kepada pihak ketiga yang tidak sah, pihak tersebut memiliki akses selama sesi berlangsung. Namun, Anda dapat segera mencabut semua izin untuk kredensial peran yang diterbitkan sebelum waktu tertentu jika perlu. Semua kredensial sementara untuk peran tersebut yang diterbitkan sebelum waktu yang ditentukan menjadi tidak berlaku. Ini memaksa semua pengguna untuk mengautentikasi ulang dan meminta kredensi baru.

> **Note**
>
> Anda tidak dapat mencabut sesi untuk peran yang terkait dengan layanan.

Saat Anda mencabut izin untuk peran menggunakan prosedur dalam topik ini, AWS lampirkan kebijakan sebaris baru ke peran yang menolak semua izin untuk semua tindakan. Ini mencakup sebuah syarat yang menerapkan pembatasan hanya jika pengguna mengasumsikan peran tersebut sebelum waktu saat Anda mencabut izin. Jika pengguna mengasumsikan peran tersebut setelah Anda mencabut izin, maka kebijakan penolakan tidak berlaku bagi pengguna tersebut.

Untuk informasi lebih lanjut tentang menolak akses, lihatMenonaktifkan izin untuk kredensial keamanan sementara.

> **Important**
>
> Kebijakan penolakan ini berlaku untuk semua pengguna peran yang ditentukan, bukan hanya untuk mereka yang memiliki sesi konsol durasi yang lebih lama.

Izin minimum untuk mencabut izin sesi dari peran

Untuk dapat berhasil mencabut izin sesi dari peran, Anda harus memiliki izin `PutRolePolicy` untuk peran tersebut. Ini memungkinkan Anda untuk melampirkan kebijakan inline `AWSRevokeOlderSessions` ke peran tersebut.

Mencabut izin sesi

Anda dapat mencabut izin sesi dari peran untuk menolak semua izin kepada pengguna mana pun yang mengambil peran tersebut.

> **Note**
>
> Anda tidak dapat mengedit peran IAM yang dibuat dari set izin Pusat IAM Identitas. Anda harus mencabut sesi set izin aktif untuk pengguna di Pusat IAM Identitas. Untuk informasi selengkapnya, lihat Mencabut sesi IAM peran aktif yang dibuat oleh set izin di Panduan Pengguna Pusat IAM Identitas.

Untuk segera menolak semua izin untuk pengguna kredensial peran saat ini

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran, lalu pilih nama (bukan kotak centang) peran yang izinnya ingin dicabut.
3. Di halaman Ringkasan untuk peran yang dipilih, pilih tab Cabut sesi.
4. Di tab Cabut sesi, pilih Cabut sesi aktif.
5. AWS meminta Anda untuk mengkonfirmasi tindakan. Pilih Saya mengakui bahwa saya mencabut semua sesi aktif untuk peran ini. centang kotak dan pilih Cabut sesi aktif pada kotak dialog.

IAMkemudian melampirkan kebijakan yang dinamai `AWSRevokeOlderSessions` untuk peran tersebut. Setelah Anda memilih Cabut sesi aktif, kebijakan menolak semua akses ke pengguna yang mengambil peran di masa lalu serta sekitar 30 detik ke depan. Pilihan future time ini memperhitungkan penundaan propagasi kebijakan untuk menangani sesi baru yang diperoleh atau diperbarui sebelum kebijakan yang diperbarui berlaku di wilayah tertentu. Setiap pengguna yang mengambil peran lebih dari sekitar 30 detik setelah Anda memilih Mencabut sesi aktif tidak terpengaruh. Untuk mempelajari mengapa perubahan tidak selalu langsung terlihat, lihat Perubahan yang saya buat tidak selalu langsung terlihat.

> **Note**
>
> Jika Anda memilih untuk Mencabut sesi aktif lagi nanti, stempel tanggal dan waktu dalam kebijakan akan di-refresh dan sekali lagi menolak semua izin untuk setiap pengguna yang mengambil peran sebelum waktu yang ditentukan.

Pengguna valid yang sesinya dicabut dengan cara ini harus memperoleh kredensial sementara untuk sesi baru untuk melanjutkan bekerja. AWS CLI Cache kredensialnya sampai kedaluwarsa. Untuk memaksa menghapus dan menyegarkan kredenal cache yang tidak lagi valid, jalankan salah satu perintah berikut: CLI

Linux, macOS, atau Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Mencabut izin sesi sebelum waktu yang ditentukan

Anda juga dapat mencabut izin sesi kapan saja sesuai pilihan Anda menggunakan AWS CLI atau SDK untuk menentukan nilai `aws: TokenIssueTime` kunci dalam elemen Kondisi kebijakan.

Kebijakan ini menolak semua izin jika nilainya lebih awal dari `aws:TokenIssueTime` tanggal dan waktu yang ditentukan. Nilai dari `aws:TokenIssueTime` sesuai dengan waktu yang tepat saat kredensial keamanan sementara dibuat. `aws:TokenIssueTime`Nilai hanya ada dalam konteks

AWS permintaan yang ditandatangani dengan kredenal keamanan sementara, sehingga pernyataan Deny dalam kebijakan tidak memengaruhi permintaan yang ditandatangani dengan kredensi jangka panjang pengguna. IAM

Kebijakan ini juga dapat dilampirkan pada suatu peran. Dalam hal ini, kebijakan hanya memengaruhi kredensial keamanan sementara yang dibuat oleh peran sebelum tanggal dan waktu yang ditentukan.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

Pengguna valid yang sesinya dicabut dengan cara ini harus memperoleh kredensial sementara untuk sesi baru untuk melanjutkan bekerja. AWS CLI Cache kredensialnya sampai kedaluwarsa. Untuk memaksa menghapus dan menyegarkan kredenal cache yang tidak lagi valid, jalankan salah satu perintah berikut: CLI

Linux, macOS, atau Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Memperbarui peran terkait layanan

Metode yang Anda gunakan untuk mengedit peran yang ditautkan ke layanan bergantung pada layanan. Beberapa layanan mungkin memungkinkan Anda mengedit izin untuk peran terkait layanan dari konsol layanan, API atau. CLI Namun, setelah Anda membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Anda dapat mengedit deskripsi peran apa pun dari IAM konsol,API, atauCLI.

Untuk informasi tentang layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait-Layanan Untuk mempelajari apakah layanan mendukung pengeditan peran terkait layanan, pilih tautan Yes untuk melihat dokumentasi peran terkait layanan untuk layanan itu.

Mengedit deskripsi peran yang ditautkan ke layanan (konsol)

Anda dapat menggunakan IAM konsol untuk mengedit deskripsi peran terkait layanan.

Untuk mengedit deskripsi peran terkait layanan (konsol)

1. Di panel navigasi IAM konsol, pilih Peran.
2. Pilih nama peran yang akan diubah.
3. Di ujung kanan Deskripsi peran, pilih Edit.
4. Masukkan deskripsi baru di kotak, lalu pilih Simpan.

Menyunting deskripsi peran terkait layanan (AWS CLI)

Anda dapat menggunakan IAM perintah dari AWS CLI untuk mengedit deskripsi peran terkait layanan.

Untuk mengubah deskripsi peran terkait layanan (AWS CLI)

1. (Opsional) Untuk melihat deskripsi peran saat ini, jalankan perintah berikut:

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   Gunakan nama peran, bukanARN, untuk merujuk ke peran dengan CLI perintah. Misalnya, jika peran memiliki yang berikutARN:`arn:aws:iam::123456789012:role/myrole`, Anda merujuk ke peran sebagai**`myrole`**.
2. Untuk memperbarui deskripsi peran terkait layanan, jalankan perintah berikut:

   ```
   aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
   ```

Mengedit deskripsi peran terkait layanan ()AWS API

Anda dapat menggunakan AWS API untuk mengedit deskripsi peran terkait layanan.

Untuk mengubah deskripsi peran terkait layanan ()AWS API

1. (Opsional) Untuk melihat deskripsi peran saat ini, panggil operasi berikut, dan sebutkan nama peran:

   AWS API: GetRole
2. Untuk memperbarui deskripsi peran, hubungi operasi berikut, dan sebutkan nama (dan deskripsi opsional) peran tersebut:

   AWS API: UpdateRole

Memperbarui kebijakan kepercayaan peran

Untuk mengubah siapa yang dapat mengasumsikan peran, Anda harus mengubah kebijakan kepercayaan peran tersebut. Anda tidak dapat mengubah kebijakan kepercayaan untuk peran yang terkait dengan layanan.

> **Catatan**
>
> - Jika pengguna terdaftar sebagai prinsipal dalam kebijakan kepercayaan peran, tetapi tidak dapat mengasumsikan peran tersebut, periksa batas izin pengguna tersebut. Jika batas izin diatur untuk pengguna tersebut, maka itu seharusnya memungkinkan tindakan `sts:AssumeRole`.
> - Untuk memungkinkan pengguna untuk mengambil peran saat ini lagi dalam sesi peran, tentukan peran ARN atau Akun AWS ARN sebagai prinsipal dalam kebijakan kepercayaan peran. Layanan AWS yang menyediakan sumber daya komputasi seperti AmazonEC2, Amazon, Amazon ECSEKS, dan Lambda memberikan kredensi sementara dan secara otomatis memperbarui kredensi ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensional yang valid. Untuk layanan ini, tidak perlu mengambil peran saat ini lagi untuk mendapatkan kredensi sementara. Namun, jika Anda berniat untuk meneruskan tag sesi atau kebijakan sesi, Anda perlu mengambil peran saat ini lagi.

Memperbarui kebijakan kepercayaan peran (konsol)

Untuk mengubah kebijakan kepercayaan peran dalam AWS Management Console

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi IAM konsol, pilih Peran.
3. Dalam daftar peran di akun Anda, pilih nama peran yang ingin Anda ubah.
4. Pilih tab Trust relationship, lalu pilih Edit trust policy.
5. Ubah kebijakan kepercayaan sebagaimana diperlukan. Untuk menambahkan prinsipal tambahan yang dapat mengasumsikan peran, tentukan dalam elemen `Principal`. Misalnya, cuplikan kebijakan berikut menunjukkan cara mereferensikan dua Akun AWS elemen: `Principal`

   ```
   "Principal": {
     "AWS": [
       "arn:aws:iam::111122223333:root",
       "arn:aws:iam::444455556666:root"
     ]
   },
   ```

   Jika Anda menetapkan prinsipal di akun lain, menambahkan akun ke kebijakan kepercayaan dari suatu peran hanya setengah dari membangun hubungan kepercayaan lintas akun. Secara default, tidak ada pengguna dalam akun terpercaya yang dapat mengasumsikan peran tersebut. Administrator untuk akun yang baru dipercaya harus memberikan izin kepada pengguna untuk mengasumsikan peran tersebut. Untuk melakukannya, administrator harus membuat atau mengubah kebijakan yang dilampirkan pada pengguna untuk memberikan akses kepada pengguna ke tindakan `sts:AssumeRole`. Untuk informasi selengkapnya, lihat prosedur berikut atau Berikan izin pengguna untuk beralih peran.

   Cuplikan kebijakan berikut menunjukkan cara mereferensikan dua AWS layanan dalam elemen: `Principal`

   ```
   "Principal": {
     "Service": [
       "opsworks.amazonaws.com",
       "ec2.amazonaws.com"
     ]
   },
   ```

6. Setelah selesai mengedit kebijakan kepercayaan Anda, pilih Perbarui kebijakan untuk menyimpan perubahan Anda.

   Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan IAMJSONreferensi elemen kebijakan.

Untuk mengizinkan pengguna dalam akun eksternal terpercaya untuk menggunakan peran (konsol)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat Berikan izin pengguna untuk beralih peran.

1. Masuk ke eksternal tepercaya Akun AWS.
2. Tentukan apakah akan melampirkan izin ke pengguna atau ke kelompok. Di panel navigasi IAM konsol, pilih Pengguna atau grup Pengguna yang sesuai.
3. Pilih nama pengguna atau kelompok yang ingin Anda beri akses, dan kemudian pilih tab Izin.
4. Lakukan salah satu hal berikut ini:

   - Untuk mengedit kebijakan yang dikelola pelanggan, pilih nama kebijakan, pilih Edit kebijakan, lalu pilih JSONtab. Anda tidak dapat mengedit kebijakan AWS terkelola. AWS kebijakan terkelola muncul dengan AWS ikon (). Untuk informasi selengkapnya tentang perbedaan antara kebijakan yang dikelola AWS dan kebijakan yang dikelola oleh pelanggan, lihat Kebijakan terkelola dan kebijakan inline.
   - Untuk mengubah kebijakan inline, pilih panah di samping nama kebijakan dan pilih Ubah Kebijakan.

5. Dalam editor kebijakan, tambahkan elemen `Statement` baru yang menentukan hal berikut:

   ```
   {
     "Effect": "Allow",
     "Action": "sts:AssumeRole",
     "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
   }
   ```

   Ganti ARN dalam pernyataan dengan ARN peran yang dapat diasumsikan pengguna.

6. Ikuti prompt pada layar untuk menyelesaikan pengubahan kebijakan.

Memperbarui kebijakan kepercayaan peran (AWS CLI)

Anda dapat menggunakan AWS CLI untuk mengubah siapa yang dapat mengambil peran.

Untuk mengubah kebijakan kepercayaan peran (AWS CLI)

1. (Opsional) Jika Anda tidak tahu nama peran yang ingin Anda ubah, jalankan perintah berikut untuk membuat daftar peran di akun Anda:

   - aws iam daftar peran

2. (Opsional) Untuk melihat penjelasan peran terkini, gunakan perintah berikut:

   - aws iam mendapatkan peran

3. Untuk mengubah prinsipal terpercaya yang dapat mengakses peran, buat file teks dengan kebijakan kepercayaan yang diperbarui. Anda dapat menggunakan editor teks apa pun untuk menyusun kebijakan.

   Misalnya, kebijakan kepercayaan berikut menunjukkan cara Akun AWS mereferensikan dua `Principal` elemen. Hal ini memungkinkan pengguna dalam dua terpisah Akun AWS untuk mengambil peran ini.

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

   Jika Anda menetapkan prinsipal di akun lain, menambahkan akun ke kebijakan kepercayaan dari suatu peran hanya setengah dari membangun hubungan kepercayaan lintas akun. Secara default, tidak ada pengguna dalam akun terpercaya yang dapat mengasumsikan peran tersebut. Administrator untuk akun yang baru dipercaya harus memberikan izin kepada pengguna untuk mengasumsikan peran tersebut. Untuk melakukannya, administrator harus membuat atau mengubah kebijakan yang dilampirkan pada pengguna untuk memberikan akses kepada

pengguna ke tindakan `sts:AssumeRole`. Untuk informasi selengkapnya, lihat prosedur berikut atau Berikan izin pengguna untuk beralih peran.

4. Untuk menggunakan file yang baru saja Anda buat untuk memperbarui kebijakan kepercayaan, jalankan perintah berikut:

   - aws iam update-assume-role-policy

Untuk mengizinkan pengguna dalam akun eksternal tepercaya untuk menggunakan peran (AWS CLI)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat Berikan izin pengguna untuk beralih peran.

1. Buat JSON file yang berisi kebijakan izin yang memberikan izin untuk mengambil peran. Misalnya, kebijakan berikut memuat izin minimum yang diperlukan:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
  }
}
```

   Ganti ARN dalam pernyataan dengan ARN peran yang dapat diasumsikan pengguna.

2. Jalankan perintah berikut untuk mengunggah JSON file yang berisi kebijakan kepercayaan kelAM:

   - aws iam membuat kebijakan

   Output dari perintah ini termasuk kebijakan. ARN Catat ini ARN karena Anda akan membutuhkannya di langkah selanjutnya.

3. Tentukan pengguna atau kelompok mana yang akan dilampiri kebijakan ini. Jika Anda tidak tahu nama pengguna atau kelompok yang dimaksudkan, gunakan salah satu perintah berikut untuk membuat daftar pengguna atau kelompok di akun Anda:

   - aws iam daftar-pengguna
   - grup daftar aws iam

4. Gunakan salah satu perintah berikut untuk melampirkan kebijakan yang Anda buat pada langkah sebelumnya kepada pengguna atau kelompok:

   - aws iam attach-user-policy
   - aws iam attach-group-policy

Memperbarui kebijakan kepercayaan peran (AWS API)

Anda dapat menggunakan AWS API untuk mengubah siapa yang dapat mengambil peran.

Untuk memodifikasi kebijakan kepercayaan peran (AWS API)

1. (Opsional) Jika Anda tidak tahu nama peran yang ingin Anda ubah, jalankan perintah berikut untuk membuat daftar peran di akun Anda:

   - ListRoles
2. (Opsional) Untuk melihat kebijakan kepercayaan terkini untuk peran, panggil operasi berikut:

   - GetRole
3. Untuk mengubah prinsipal terpercaya yang dapat mengakses peran, buat file teks dengan kebijakan kepercayaan yang diperbarui. Anda dapat menggunakan editor teks apa pun untuk menyusun kebijakan.

   Misalnya, kebijakan kepercayaan berikut menunjukkan cara Akun AWS mereferensikan dua `Principal` elemen. Hal ini memungkinkan pengguna dalam dua terpisah Akun AWS untuk mengambil peran ini.

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

Jika Anda menetapkan prinsipal di akun lain, menambahkan akun ke kebijakan kepercayaan dari suatu peran hanya setengah dari membangun hubungan kepercayaan lintas akun. Secara default, tidak ada pengguna dalam akun terpercaya yang dapat mengasumsikan peran tersebut. Administrator untuk akun yang baru dipercaya harus memberikan izin kepada pengguna untuk mengasumsikan peran tersebut. Untuk melakukannya, administrator harus membuat atau mengubah kebijakan yang dilampirkan pada pengguna untuk memberikan akses kepada pengguna ke tindakan `sts:AssumeRole`. Untuk informasi selengkapnya, lihat prosedur berikut atau Berikan izin pengguna untuk beralih peran.

4. Untuk menggunakan file yang baru saja Anda buat untuk memperbarui kebijakan kepercayaan, panggil operasi berikut:

   - UpdateAssumeRolePolicy

Untuk memungkinkan pengguna di akun eksternal tepercaya menggunakan role (AWS API)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat Berikan izin pengguna untuk beralih peran.

1. Buat JSON file yang berisi kebijakan izin yang memberikan izin untuk mengambil peran. Misalnya, kebijakan berikut memuat izin minimum yang diperlukan:

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
     }
   }
   ```

   Ganti ARN dalam pernyataan dengan ARN peran yang dapat diasumsikan pengguna.

2. Panggil operasi berikut untuk mengunggah JSON file yang berisi kebijakan kepercayaan keIAM:

   - CreatePolicy

   Output dari operasi ini termasuk kebijakan. ARN Catat ini ARN karena Anda akan membutuhkannya di langkah selanjutnya.

3. Tentukan pengguna atau kelompok mana yang akan dilampiri kebijakan ini. Jika Anda tidak tahu nama pengguna atau grup yang dimaksudkan, panggil salah satu operasi berikut untuk membuat daftar pengguna atau kelompok di akun Anda:

   - ListUsers
   - ListGroups
4. Panggil salah satu perintah berikut untuk melampirkan kebijakan yang Anda buat pada langkah sebelumnya kepada pengguna atau kelompok:

   - API: AttachUserPolicy
   - AttachGroupPolicy

Memperbarui izin untuk peran

Gunakan prosedur berikut untuk memperbarui kebijakan izin peran dan batasan izin.

Prasyarat: Lihat akses peran

Sebelum mengubah izin pengguna, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Memperbarui kebijakan izin untuk peran

Untuk mengubah izin yang diperbolehkan oleh peran tersebut, ubah kebijakan (atau beberapa kebijakan) izin peran tersebut. Anda tidak dapat mengubah kebijakan izin untuk peran terkait layanan. IAM Anda mungkin dapat mengubah kebijakan izin dalam layanan yang bergantung pada peran tersebut. Untuk memeriksa apakah layanan mendukung fitur ini, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya dalam kolom Peran yang terkait dengan layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Memperbarui kebijakan izin peran (konsol)

Untuk mengubah izin yang diperbolehkan oleh peran (konsol)

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi IAM konsol, pilih Peran.

3. Pilih nama peran yang ingin Anda ubah, dan kemudian pilih tabIzin.
4. Lakukan salah satu langkah berikut:

   - Untuk mengubah kebijakan yang dikelola oleh pelanggan yang sudah ada, pilih nama kebijakan tersebut dan kemudian pilih Ubah kebijakan.

     > **Note**
     >
     > Anda tidak dapat mengedit kebijakan AWS terkelola.
     > AWS kebijakan terkelola muncul dengan AWS ikon
     > ().
     > Untuk informasi selengkapnya tentang perbedaan antara kebijakan AWS terkelola dan kebijakan yang dikelola pelanggan, lihatKebijakan terkelola dan kebijakan inline.

   - Untuk melampirkan kebijakan terkelola yang ada ke peran, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.
   - Untuk mengedit kebijakan inline yang ada, perluas kebijakan dan pilih Edit.
   - Untuk menyematkan kebijakan sebaris baru, pilih Tambahkan izin, lalu pilih Buat kebijakan sebaris.
   - Untuk menghapus kebijakan yang ada dari peran, pilih kotak centang di samping nama kebijakan, lalu pilih Hapus.

Memperbarui kebijakan izin peran ()AWS CLI

Untuk mengubah izin yang diperbolehkan oleh peran tersebut, ubah kebijakan (atau beberapa kebijakan) izin peran tersebut. Anda tidak dapat mengubah kebijakan izin untuk peran terkait layanan. IAM Anda mungkin dapat mengubah kebijakan izin dalam layanan yang bergantung pada peran tersebut. Untuk memeriksa apakah layanan mendukung fitur ini, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya dalam kolom Peran yang terkait dengan layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Untuk mengubah izin yang diperbolehkan oleh peran (AWS CLI)

1. (Opsional) Untuk melihat izin yang saat ini terkait dengan peran, jalankan perintah berikut:

   1. aws iam list-role-policies untuk mencantumkan kebijakan sebaris

   2. aws iam list-attached-role -policies untuk membuat daftar kebijakan terkelola
2. Perintah untuk memperbarui izin untuk peran berbeda-beda tergantung pada apakah Anda memperbarui kebijakan terkelola atau kebijakan inline.

   Untuk memperbarui kebijakan terkelola, panggil operasi berikut untuk membuat versi baru kebijakan terkelola:

   - aws iam create-policy-version

   Untuk memperbarui kebijakan inline, jalankan perintah berikut:

   - aws iam put-role-policy

Memperbarui kebijakan izin peran ()AWS API

Untuk mengubah izin yang diperbolehkan oleh peran tersebut, ubah kebijakan (atau beberapa kebijakan) izin peran tersebut. Anda tidak dapat mengubah kebijakan izin untuk peran terkait layanan. IAM Anda mungkin dapat mengubah kebijakan izin dalam layanan yang bergantung pada peran tersebut. Untuk memeriksa apakah layanan mendukung fitur ini, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya dalam kolom Peran yang terkait dengan layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Untuk mengubah izin yang diizinkan oleh role ()AWS API

1. (Opsional) Untuk melihat izin yang saat ini terkait dengan peran, panggil perintah berikut:

   1. ListRolePolicies untuk membuat daftar kebijakan inline
   2. ListAttachedRolePolicies untuk membuat daftar kebijakan terkelola
2. Operasi untuk memperbarui izin untuk peran berbeda-beda tergantung pada apakah Anda memperbarui kebijakan terkelola atau kebijakan inline.

   Untuk memperbarui kebijakan terkelola, panggil operasi berikut untuk membuat versi baru kebijakan terkelola:

   - CreatePolicyVersion

   Untuk memperbarui kebijakan inline, panggil operasi berikut:

- PutRolePolicy

Perbarui batas izin untuk peran

Untuk mengubah izin maksimum yang diperbolehkan untuk peran, ubah batas izin peran.

Memperbarui batas izin peran (konsol)

Untuk mengubah kebijakan yang digunakan untuk mengatur batas izin untuk peran

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Pilih nama peran dengan batas izin yang ingin Anda ubah.
4. Pilih tab Izin. Jika perlu, buka bagian Batas izin lalu pilih Ubah batas.
5. Pilih kebijakan yang ingin Anda gunakan untuk batasan izin.
6. Pilih Ubah batas.

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini.

Memperbarui batas izin peran ()AWS CLI

Untuk mengubah kebijakan terkelola yang digunakan untuk mengatur batas izin peran (AWS CLI)

1. (Opsional) Untuk melihat batas izin terkini untuk peran, jalankan perintah berikut:

   - aws iam mendapatkan peran
2. Untuk menggunakan kebijakan terkelola yang berbeda untuk memperbarui batas izin untuk peran, jalankan perintah berikut:

   - batas aws iam put-role-permissions

   Peran hanya dapat memiliki satu kebijakan terkelola yang diatur sebagai batas izin. Jika Anda mengubah batas izin, Anda juga mengubah izin maksimum yang diperbolehkan untuk peran.

Memperbarui batas izin peran ()AWS API

Untuk mengubah kebijakan terkelola yang digunakan untuk menyetel batas izin untuk peran ()AWS API

1. (Opsional) Untuk melihat batas izin terkini untuk peran, panggil operasi berikut:
   - GetRole
2. Untuk menggunakan kebijakan terkelola untuk memperbarui batas izin untuk peran, panggil operasi berikut:
   - PutRolePermissionsBoundary

   Peran hanya dapat memiliki satu kebijakan terkelola yang diatur sebagai batas izin. Jika Anda mengubah batas izin, Anda juga mengubah izin maksimum yang diperbolehkan untuk peran.

Perbarui setelan untuk peran

Gunakan prosedur berikut untuk memperbarui deskripsi peran atau mengubah durasi sesi maksimum untuk peran.

Memperbarui deskripsi peran

Untuk mengubah deskripsi peran, ubah teks deskripsi.

Memperbarui deskripsi peran (konsol)

Untuk mengubah deskripsi peran (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi IAM konsol, pilih Peran.
3. Pilih nama peran yang akan diubah.
4. Di bagian Ringkasan, pilih Edit.
5. Memasukkan Deskripsi baru di kotak, dan memilih Simpan perubahan.

Memperbarui deskripsi peran (AWS CLI)

Untuk mengubah deskripsi peran (AWS CLI)

1. (Opsional) Untuk melihat deskripsi peran terkini, jalankan perintah berikut:
   - aws iam mendapatkan peran
2. Untuk memperbarui deskripsi peran, jalankan perintah berikut dengan parameter deskripsi:
   - peran pembaruan aws iam

Memperbarui deskripsi peran (AWS API)

Untuk mengubah deskripsi peran (AWS API)

1. (Opsional) Untuk melihat desripsi terkini untuk peran, panggil operasi berikut:
   - GetRole
2. Untuk memperbarui deskripsi peran, panggil operasi berikut dengan parameter deskripsi:
   - UpdateRole

Memperbarui durasi sesi maksimum untuk peran

Untuk menentukan pengaturan durasi sesi maksimum untuk peran yang diasumsikan menggunakan konsol, AWS CLI, atau AWS API, ubah nilai pengaturan durasi sesi maksimum. Pengaturan ini dapat memiliki nilai dari 1 jam hingga 12 jam. Jika Anda tidak menentukan nilai, defaultnya maksimum 1 jam diterapkan. Pengaturan ini tidak membatasi sesi yang diasumsikan oleh AWS layanan.

Memperbarui durasi sesi peran maksimum (konsol)

Untuk mengubah setelan durasi sesi maksimum untuk peran yang diasumsikan menggunakan konsol, AWS CLI, atau AWS API (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi IAM konsol, pilih Peran.
3. Pilih nama peran yang akan diubah.
4. Di bagian Ringkasan, pilih Edit.

5. Untuk Durasi sesi maksimum, pilih nilai. Atau, pilih Durasi kustom dan masukkan nilai (dalam detik).
6. Pilih Simpan perubahan.

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini. Untuk mempelajari cara mencabut sesi yang ada untuk peran ini, lihat Mencabut kredensi IAM keamanan sementara peran.

Dalam AWS Management Console, sesi IAM pengguna adalah 12 jam secara default. IAMpengguna yang beralih peran di konsol diberikan durasi sesi maksimum peran, atau sisa waktu dalam sesi pengguna, mana yang lebih sedikit.

Siapa pun yang mengambil peran dari AWS CLI atau AWS API dapat meminta sesi yang lebih lama, hingga maksimum ini. Pengaturan `MaxSessionDuration` menentukan durasi maksimum dari sesi peran yang dapat diminta.

- Untuk menentukan durasi sesi menggunakan AWS CLI menggunakan `duration-seconds` parameter. Untuk mempelajari selengkapnya, lihat Beralih ke IAM peran (AWS CLI).
- Untuk menentukan durasi sesi menggunakan AWS API, gunakan `DurationSeconds` parameter. Untuk mempelajari selengkapnya, lihat Beralih ke IAM peran (AWS API).

Memperbarui durasi sesi peran maksimum (AWS CLI)

> **Note**
>
> Siapa pun yang mengambil peran dari AWS CLI atau API dapat menggunakan `duration-seconds` CLI parameter atau `DurationSeconds` API parameter untuk meminta sesi yang lebih lama. Pengaturan `MaxSessionDuration` menentukan durasi maksimum dari sesi peran yang dapat diminta dengan menggunakan parameter `DurationSeconds`. Jika pengguna tidak menentukan nilai untuk parameter `DurationSeconds` tersebut, kredensial keamanan mereka berlaku selama satu jam.

Untuk mengubah setelan durasi sesi maksimum untuk peran yang diasumsikan menggunakan AWS CLI (AWS CLI)

1. (Opsional) Untuk melihat durasi sesi maksimum peran untuk peran, jalankan perintah berikut:

- aws iam mendapatkan peran

2. Untuk memperbarui pengaturan durasi sesi maksimum peran, jalankan perintah berikut dengan `max-session-duration` CLI parameter atau `MaxSessionDuration` API parameter:

   - peran pembaruan aws iam

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini. Untuk mempelajari cara mencabut sesi yang ada untuk peran ini, lihat Mencabut kredensi IAM keamanan sementara peran.

Memperbarui durasi sesi peran maksimum (AWS API)

> **Note**
>
> Siapa pun yang mengambil peran dari AWS CLI atau API dapat menggunakan `duration-seconds` CLI parameter atau `DurationSeconds` API parameter untuk meminta sesi yang lebih lama. Pengaturan `MaxSessionDuration` menentukan durasi maksimum dari sesi peran yang dapat diminta dengan menggunakan parameter `DurationSeconds`. Jika pengguna tidak menentukan nilai untuk parameter `DurationSeconds` tersebut, kredensial keamanan mereka berlaku selama satu jam.

Untuk mengubah setelan durasi sesi maksimum untuk peran yang diasumsikan menggunakan API (AWS API)

1. (Opsional) Untuk melihat durasi sesi maksimum terkini untuk peran, panggil operasi berikut:

   - GetRole

2. Untuk memperbarui setelan durasi sesi maksimum peran, panggil operasi berikut dengan `max-sessionduration` CLI parameter atau `MaxSessionDuration` API parameter:

   - UpdateRole

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini. Untuk mempelajari cara mencabut sesi yang ada untuk peran ini, lihat Mencabut kredensi IAM keamanan sementara peran.

Hapus peran atau profil contoh

Jika Anda tidak lagi memerlukan peran, kami sarankan Anda untuk menghapus peran tersebut dan izin terkaitnya. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif.

Jika peran dikaitkan dengan EC2 instance, Anda juga dapat menghapus peran tersebut dari profil instance dan kemudian menghapus profil instance.

> **Warning**
>
> Pastikan Anda tidak memiliki EC2 instans Amazon yang berjalan dengan profil peran atau instance yang akan Anda hapus. Menghapus peran atau profil instans yang terkait dengan suatu instans yang sedang berjalan akan merusak aplikasi yang sedang berjalan pada instans tersebut.

Jika Anda memilih untuk tidak menghapus peran secara permanen, Anda dapat menonaktifkan peran. Untuk melakukannya, ubah kebijakan peran lalu cabut semua sesi saat ini. Misalnya, Anda dapat menambahkan kebijakan ke peran yang menolak akses ke semua peran AWS. Anda juga dapat mengubah kebijakan kepercayaan untuk menolak akses ke siapa pun yang mencoba untuk mengasumsikan peran tersebut. Untuk informasi lebih lanjut tentang mencabut sesi, lihat Mencabut kredensi IAM keamanan sementara peran.

Topik

- Melihat akses peran
- Menghapus peran yang terkait dengan layanan
- Menghapus peran IAM (konsol)
- Menghapus peran IAM (AWS CLI)
- Menghapus IAM peran ()AWS API
- Informasi terkait

Melihat akses peran

Sebelum menghapus peran, kami menyarankan agar Anda meninjau kapan peran tersebut terakhir kali digunakan. Anda dapat melakukan ini menggunakan AWS Management Console, the AWS CLI,

atau AWS API. Anda harus melihat informasi ini karena Anda tidak ingin menghapus akses dari seseorang yang menggunakan peran tersebut.

Tanggal aktivitas terakhir peran mungkin tidak cocok dengan tanggal terakhir yang dilaporkan di tab Terakhir Diakses. Tab Terakhir Diakses melaporkan aktivitas hanya untuk layanan yang diizinkan oleh kebijakan izin peran. Tanggal aktivitas terakhir peran mencakup upaya terakhir untuk mengakses layanan apa pun AWS.

> **Note**
>
> Periode pelacakan untuk aktivitas terakhir peran dan data Terakhir Diakses adalah selama 400 hari terakhir. Periode ini bisa lebih singkat jika Wilayah Anda mulai mendukung fitur-fitur ini tahun lalu. Peran ini mungkin telah digunakan lebih dari 400 hari yang lalu. Untuk informasi lebih lanjut tentang periode pelacakan ini, lihat Tempat AWS melacak informasi terakhir yang diakses.

Untuk melihat kapan peran terakhir digunakan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Cari baris peran dengan aktivitas ingin Anda lihat. Anda dapat menggunakan bidang pencarian untuk mempersempit hasil. Lihat kolom Aktivitas terakhir untuk melihat jumlah hari sejak peran terakhir digunakan. Jika peran tersebut belum digunakan dalam periode pelacakan, maka tabel akan menampilkan Tidak ada.
4. Pilih nama peran untuk melihat informasi lebih lanjut. Halaman Ringkasan peran juga mencakup Aktivitas terakhir, yang menampilkan tanggal terakhir peran digunakan. Jika peran tersebut tidak digunakan dalam 400 hari terakhir, maka Aktivitas terakhir menampilkan Tidak diakses dalam periode pelacakan.

Untuk menampilkan waktu terakhir peran digunakan (AWS CLI)

`aws iam get-role` - Jalankan perintah ini untuk mengembalikan informasi tentang peran, termasuk objek`RoleLastUsed`. Objek ini berisi `LastUsedDate` dan `Region` yang perannya terakhir digunakan. Jika `RoleLastUsed` ada, tetapi tidak berisi nilai, maka peran tersebut belum pernah digunakan dalam periode pelacakan.

Untuk melihat kapan peran terakhir digunakan (AWS API)

`GetRole` - Panggil operasi ini untuk mengembalikan informasi tentang peran, termasuk objek`RoleLastUsed`. Objek ini berisi `LastUsedDate` dan `Region` yang perannya terakhir digunakan. Jika `RoleLastUsed` ada, tetapi tidak berisi nilai, maka peran tersebut belum pernah digunakan dalam periode pelacakan.

Menghapus peran yang terkait dengan layanan

Metode yang Anda gunakan untuk menghapus peran terkait layanan bergantung pada layanan. Dalam beberapa kasus, Anda tidak perlu menghapus peran terkait layanan secara manual. Misalnya, ketika Anda menyelesaikan tindakan tertentu (seperti menghapus sumber daya) dalam layanan, layanan mungkin akan menghapus peran terkait layanan untuk Anda. Dalam kasus lain, layanan mungkin mendukung penghapusan peran terkait layanan secara manual dari konsol layanan, API atau. AWS CLI

Tinjau dokumentasi untuk peran terkait layanan di layanan tertaut untuk mempelajari cara menghapus peran. Anda dapat melihat peran terkait layanan di akun Anda dengan membuka halaman IAM Peran di konsol. Peran yang terkait dengan layanan muncul dengan (Peran yang terkait dengan layanan) dalam kolom Entitas terpercaya di tabel. Banner pada halaman Ringkasan peran juga menunjukkan bahwa peran tersebut adalah peran yang terkait dengan layanan.

Jika layanan tidak menyertakan dokumentasi untuk menghapus peran terkait layanan, Anda dapat menggunakan IAM konsol AWS CLI, atau API menghapus peran.

Menghapus peran IAM (konsol)

Saat Anda menggunakan AWS Management Console untuk menghapus peran, IAM secara otomatis melepaskan kebijakan terkelola yang terkait dengan peran tersebut. Ini juga secara otomatis menghapus kebijakan sebaris apa pun yang terkait dengan peran, dan profil EC2 instans Amazon apa pun yang berisi peran tersebut.

> **Important**
>
> Dalam beberapa kasus, peran mungkin dikaitkan dengan profil EC2 instans Amazon, dan peran serta profil instance mungkin memiliki nama yang sama. Dalam hal ini Anda dapat menggunakan AWS Management Console untuk menghapus peran dan profil instance. Pertautan ini terjadi secara otomatis untuk peran dan profil instans yang Anda buat di konsol. Jika Anda membuat peran dari AWS CLI, Alat untuk Windows PowerShell, atau AWS API,

maka peran dan profil instans mungkin memiliki nama yang berbeda. Jika demikian, Anda tidak dapat menggunakan konsol untuk menghapusnya. Sebagai gantinya, Anda harus menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk terlebih dahulu menghapus peran dari profil instance. Kemudian, Anda harus melakukan langkah terpisah untuk menghapus peran tersebut.

Untuk menghapus peran (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran, lalu centang kotak di samping nama peran yang ingin Anda hapus.
3. Pilih Hapus di bagian atas halaman.
4. Di kotak dialog konfirmasi, tinjau informasi yang terakhir diakses, yang menunjukkan kapan masing-masing peran yang dipilih terakhir mengakses AWS layanan. Ini membantu Anda mengonfirmasi jika peran tersebut saat ini aktif. Jika Anda ingin melanjutkan, masukkan nama peran di kolom input teks dan pilih Hapus. Jika Anda yakin, Anda dapat melanjutkan dengan penghapusan meskipun informasi yang terakhir diakses masih dimuat.

> **Note**
>
> Anda tidak dapat menggunakan konsol untuk menghapus profil instans kecuali itu memiliki nama yang sama dengan peran. Profil instans dihapus sebagai bagian dari proses penghapusan peran sebagaimana dijelaskan dalam prosedur sebelumnya. Untuk menghapus profil instance tanpa juga menghapus peran, Anda harus menggunakan AWS CLI atau AWS API. Untuk informasi selengkapnya, silakan lihat bagian-bagian berikut ini.

Menghapus peran IAM (AWS CLI)

Saat Anda menggunakan AWS CLI untuk menghapus peran, Anda harus terlebih dahulu menghapus kebijakan sebaris yang terkait dengan peran tersebut. Anda juga harus melepaskan kebijakan terkelola yang terkait dengan peran tersebut. Jika Anda ingin menghapus profil instance terkait yang berisi peran, Anda harus menghapusnya secara terpisah.

Untuk menghapus peran (AWS CLI)

1. Jika Anda tidak tahu nama peran yang ingin Anda jalankan, jalankan perintah berikut untuk membuat daftar peran di akun Anda:

   ```
   aws iam list-roles
   ```

   Daftar ini mencakup Amazon Resource Name (ARN) dari setiap peran. Gunakan nama peran, bukanARN, untuk merujuk ke peran dengan CLI perintah. Misalnya, jika peran memiliki yang berikutARN:`arn:aws:iam::123456789012:role/myrole`, Anda merujuk ke peran sebagai**`myrole`**.

2. Hapus peran dari semua profil instance yang terkait dengan peran tersebut.

   a. Untuk membuat daftar semua profil instans berkaitan dengan peran tersebut, masukkan perintah berikut:

      ```
      aws iam list-instance-profiles-for-role --role-name role-name
      ```

   b. Untuk menghapus peran dari profil instans, masukkan perintah berikut untuk setiap profil instans:

      ```
      aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
      ```

3. Hapus semua kebijakan yang terkait dengan peran tersebut.

   a. Untuk mencantumkan semua kebijakan inline yang ada dalam peran, masukkan perintah berikut:

      ```
      aws iam list-role-policies --role-name role-name
      ```

   b. Untuk menghapus setiap kebijakan sebaris dari peran, masukkan perintah berikut untuk setiap kebijakan:

      ```
      aws iam delete-role-policy --role-name role-name --policy-name policy-name
      ```

   c. Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke peran, masukkan perintah berikut:

```
aws iam list-attached-role-policies --role-name role-name
```

d. Untuk melepaskan setiap kebijakan terkelola dari peran, masukkan perintah berikut untuk setiap kebijakan:

```
aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
```

4. Masukkan perintah berikut untuk menghapus peran:

```
aws iam delete-role --role-name role-name
```

5. Jika Anda tidak berencana untuk menggunakan ulang profil instans yang terkait dengan peran tersebut, Anda dapat memasukkan perintah berikut untuk menghapusnya:

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Menghapus IAM peran ()AWS API

Saat Anda menggunakan IAM API untuk menghapus peran, Anda harus terlebih dahulu menghapus kebijakan sebaris yang terkait dengan peran tersebut. Anda juga harus melepaskan kebijakan terkelola yang terkait dengan peran tersebut. Jika Anda ingin menghapus profil instance terkait yang berisi peran, Anda harus menghapusnya secara terpisah.

Untuk menghapus peran (AWS API)

1. Untuk mencantumkan semua profil instance yang terkait dengan peran, hubungi ListInstanceProfilesForRole.

   Untuk menghapus peran dari profil instance, panggil RemoveRoleFromInstanceProfile. Anda harus meneruskan nama peran dan nama profil instans.

   Jika Anda tidak akan menggunakan kembali profil instance yang dikaitkan dengan peran tersebut, panggil DeleteInstanceProfileuntuk menghapusnya.

2. Untuk mencantumkan semua kebijakan sebaris untuk peran, hubungi ListRolePolicies.

   Untuk menghapus kebijakan sebaris yang terkait dengan peran, panggil DeleteRolePolicy. Anda harus meneruskan nama peran dan nama kebijakan sebaris.

3. Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke peran, hubungi ListAttachedRolePolicies.

   Untuk melepaskan kebijakan terkelola yang melekat pada peran, hubungi DetachRolePolicy. Anda harus meneruskan nama peran dan kebijakan terkelolaARN.

4. Panggil DeleteRole untuk menghapus peran.

Informasi terkait

Untuk informasi umum tentang profil instans, lihat Gunakan profil contoh.

Untuk informasi umum tentang peran yang terkait dengan layanan, lihat Buat peran tertaut layanan.

Metode untuk mengambil peran

Sebelum pengguna, aplikasi, atau layanan dapat menggunakan peran yang Anda buat, Anda harus memberikan izin untuk beralih ke peran tersebut. Anda dapat menggunakan kebijakan apa pun yang dilampirkan ke grup atau pengguna untuk memberikan izin yang diperlukan. Setelah izin diberikan, pengguna dapat mengambil peran dari AWS Management Console, Alat untuk Windows PowerShell, AWS Command Line Interface (AWS CLI) dan. `AssumeRole`API

> **Important**
>
> Saat Anda membuat peran secara terprogram alih-alih di IAM konsol, Anda memiliki opsi untuk menambahkan hingga 512 karakter selain`RoleName`, yang panjangnya bisa mencapai 64 karakter. `Path` Namun, jika Anda bermaksud menggunakan peran dengan fitur Switch Role di AWS Management Console, maka gabungan `Path` dan `RoleName` tidak dapat melebihi 64 karakter.

Metode yang digunakan untuk mengambil peran menentukan siapa yang dapat mengambil peran dan berapa lama sesi peran dapat berlangsung. Saat menggunakan `AssumeRole*` API operasi, IAM peran yang Anda asumsikan adalah sumber daya. Pengguna atau peran yang memanggil `AssumeRole*` API operasi adalah prinsipal.

Tabel berikut membandingkan metode untuk mengasumsikan peran.

Metode untuk mengasumsikan peran	Siapa yang bisa mengambil peran	Metode untuk menentukan masa pakai kredensi	Masa pakai kredensi (min | maks | default)
AWS Management Console	Pengguna (dengan beralih peran)	Durasi sesi maksimum pada Halaman ringkasan Peran	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
`assume-role` CLIatau `AssumeRole` APIoperasi	Pengguna atau peran[1]	`duration-seconds` CLIatau `DurationSeconds` API parameter	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
`assume-role-with-saml` CLIatau `AssumeRoleWithSAML` APIoperasi	Setiap pengguna yang diautentikasi menggunakan SAML	`duration-seconds` CLIatau `DurationSeconds` API parameter	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
`assume-role-with-web-identity` CLIatau `AssumeRoleWithWebIdentity` APIoperasi	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	`duration-seconds` CLIatau `DurationSeconds` API parameter	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
Konsol URL dibangun dengan `AssumeRole`	Pengguna atau peran	`SessionDuration` HTMLparameter di URL	15 menit | 12 jam | 1 jam

Metode untuk mengasumsikan peran	Siapa yang bisa mengambil peran	Metode untuk menentukan masa pakai kredensi	Masa pakai kredensi (min | maks | default)
Konsol URL dibangun dengan `AssumeRoleWithSAML`	Setiap pengguna yang diautentikasi menggunakan SAML	`SessionDuration` HTMLparameter di URL	15 menit | 12 jam | 1 jam
Konsol URL dibangun dengan `AssumeRoleWithWebIdentity`	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	`SessionDuration` HTMLparameter di URL	15 menit | 12 jam | 1 jam

[1] Menggunakan kredensial untuk satu peran untuk mengasumsikan peran berbeda disebut rantai peran. Saat Anda menggunakan rantai peran, kredensial baru Anda dibatasi hingga durasi maksimum satu jam. Bila Anda menggunakan peran untuk memberikan izin ke aplikasi yang berjalan pada EC2 instance, aplikasi tersebut tidak tunduk pada batasan ini.

[2] Pengaturan ini dapat memiliki nilai dari 1 jam hingga 12 jam. Untuk detail tentang pengubahan pengaturan durasi sesi maksimum, lihat IAMmanajemen peran. Setelan ini menentukan durasi sesi maksimum yang dapat Anda minta saat Anda mendapatkan kredensial peran. Misalnya, saat Anda menggunakan AssumeRoleAPIoperasi* untuk mengambil peran, Anda dapat menentukan panjang sesi menggunakan `DurationSeconds` parameter. Gunakan parameter ini untuk menentukan panjang durasi sesi peran mulai dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. IAMpengguna yang beralih peran di konsol diberikan durasi sesi maksimum, atau sisa waktu dalam sesi pengguna mereka, mana yang kurang. Anggaplah Anda menetapkan durasi maksimum 5 jam dalam suatu peran. IAMPengguna yang telah masuk ke konsol selama 10 jam (dari maksimum default 12) beralih ke peran. Durasi sesi peran yang tersedia adalah 2 jam. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran nanti di halaman ini.

Catatan

- Pengaturan durasi sesi maksimum tidak membatasi sesi yang diambil oleh layanan AWS .
- Kredensi EC2 IAM peran Amazon tidak tunduk pada durasi sesi maksimum yang dikonfigurasi dalam peran.
- Untuk memungkinkan pengguna untuk mengambil peran saat ini lagi dalam sesi peran, tentukan peran ARN atau Akun AWS ARN sebagai prinsipal dalam kebijakan kepercayaan peran. Layanan AWS yang menyediakan sumber daya komputasi seperti AmazonEC2, Amazon, Amazon ECSEKS, dan Lambda memberikan kredensi sementara dan secara otomatis memperbarui kredensi ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensional yang valid. Untuk layanan ini, tidak perlu mengambil peran saat ini lagi untuk mendapatkan kredensi sementara. Namun, jika Anda berniat untuk meneruskan tag sesi atau kebijakan sesi, Anda perlu mengambil peran saat ini lagi. Untuk mempelajari cara memodifikasi kebijakan kepercayaan peran untuk menambahkan peran utama ARN atau Akun AWS ARN, lihatMemperbarui kebijakan kepercayaan peran .

Topik

- Beralih dari pengguna ke IAM peran (konsol)
- Beralih ke IAM peran (AWS CLI)
- Beralih ke IAM peran (Alat untuk Windows PowerShell)
- Beralih ke IAM peran (AWS API)
- Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2
- Gunakan profil contoh

Beralih dari pengguna ke IAM peran (konsol)

Anda dapat beralih peran saat masuk sebagai IAM pengguna, pengguna di Pusat IAM Identitas, peran SAML -federasi, atau peran federasi identitas web. Peran menentukan sekumpulan izin yang dapat Anda gunakan untuk mengakses AWS sumber daya yang Anda butuhkan. Namun, Anda tidak masuk ke peran, tetapi setelah masuk sebagai IAM pengguna, Anda dapat beralih ke IAM peran. Sementara waktu ini mengesampingkan izin pengguna awal Anda dan justru memberikan izin yang ditetapkan untuk peran. Perannya bisa di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat IAMperan, dan IAMpenciptaan peran.

Izin pengguna Anda dan peran apa pun yang Anda alihkan tidak bersifat kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda beralih ke suatu peran, sementara Anda meninggalkan izin pengguna dan bekerja dengan izin yang diberikan ke peran tersebut. Saat Anda keluar dari peran, izin pengguna asli Anda dipulihkan secara otomatis.

Saat Anda beralih peran AWS Management Console, konsol selalu menggunakan kredenal asli Anda untuk mengotorisasi sakelar. Misalnya, jika Anda beralih ke RoleAIAM, gunakan kredensi asli Anda untuk menentukan apakah Anda diizinkan untuk mengasumsikan RoleA. Jika Anda kemudian beralih ke RoleB saat Anda menggunakan RoleA, masih menggunakan kredensi asli Anda untuk mengotorisasi sakelar AWS , bukan kredenal untuk RoleA.

> **Note**
>
> Saat Anda masuk sebagai pengguna di Pusat IAM Identitas, sebagai peran SAML -federasi, atau sebagai peran federasi identitas web, Anda akan mengambil IAM peran saat memulai sesi. Misalnya, ketika pengguna di Pusat IAM Identitas masuk ke portal AWS akses, mereka harus memilih set izin yang berkorelasi dengan peran sebelum mereka dapat mengakses AWS sumber daya.

Sesi peran

Saat Anda beralih peran, AWS Management Console sesi Anda berlangsung selama 1 jam secara default. IAMsesi pengguna 12 jam secara default, pengguna lain mungkin memiliki durasi sesi yang berbeda ditentukan. Saat Anda beralih peran di konsol, Anda akan diberikan durasi sesi maksimum peran, atau sisa waktu di sesi pengguna, mana yang lebih sedikit. Anda tidak dapat memperpanjang durasi sesi dengan mengambil peran. Misalnya, anggaplah durasi sesi maksimum 10 jam ditetapkan untuk peran. Anda telah masuk ke konsol selama 8 jam ketika Anda memutuskan untuk beralih ke peran. Ada 4 jam tersisa di sesi pengguna Anda, jadi durasi sesi peran yang diizinkan adalah 4 jam, bukan durasi sesi maksimum 10 jam. Tabel berikut menunjukkan cara menentukan durasi sesi untuk IAM pengguna saat beralih peran di konsol.

IAMwaktu sesi pengguna yang tersisa adalah...	Durasi sesi peran adalah..		
Kurang dari durasi sesi maksimum peran	Waktu yang tersisa di sesi pengguna		
Lebih dari durasi sesi maksimum peran	Nilai durasi sesi maksimum		
Setara dengan durasi sesi maksimum peran	Nilai durasi sesi maksimum (perkiraan)		

> **Note**
>
> Beberapa konsol AWS layanan dapat memperbarui sesi peran Anda secara otomatis ketika berakhir tanpa Anda mengambil tindakan apa pun. Beberapa mungkin akan meminta Anda memuat ulang halaman peramban untuk mengotentikasi ulang sesi Anda.

Pertimbangan

- Anda tidak dapat beralih peran jika Anda masuk sebagai Pengguna root akun AWS.
- Pengguna harus diberikan izin untuk beralih peran berdasarkan kebijakan. Untuk petunjuk, silakan lihat Berikan izin pengguna untuk beralih peran.
- Anda tidak dapat mengganti peran AWS Management Console ke peran yang membutuhkan ExternalIdnilai. Anda dapat beralih ke peran seperti itu hanya dengan memanggil `AssumeRole`APIyang mendukung `ExternalId` parameter.

Untuk beralih ke peran

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.

2. Di halaman Beranda Konsol, pilih IAM layanan.
3. Di AWS Management Console, pilih nama pengguna Anda di bilah navigasi di kanan atas. Itu seperti ini: ***username***@***account_ID_number_or_alias***.
4. Pilih Ganti Peran.
5. Pada halaman Ganti Peran, ketik nomor ID akun atau alias akun dan nama peran yang diberikan oleh administrator Anda.

   > **Note**
   >
   > Jika administrator Anda membuat peran itu dengan sebuah jalur, seperti `division_abc/subdivision_efg/roleToDoX`, maka Anda harus mengetik jalur dan nama lengkap tersebut di kotak Peran. Jika Anda hanya mengetik nama peran, atau jika gabungan `Path` dan `RoleName` lebih dari 64 karakter, pergantian peran gagal. Ini adalah batasan cookie peramban yang menyimpan nama peran tersebut. Jika ini terjadi, hubungi administrator Anda dan minta mereka untuk mengurangi ukuran jalur dan nama peran.

6. (Opsional) Anda dapat memasukkan nama tampilan dan memilih warna tampilan yang akan menyoroti peran di bilah navigasi konsol.

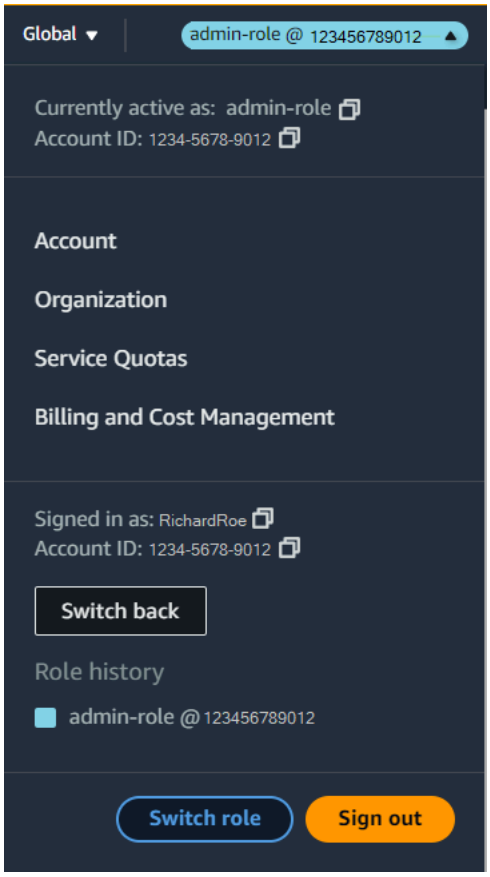
   - Untuk Menampilkan jenis nama teks yang ingin Anda tampilkan di bilah navigasi sebagai pengganti nama pengguna Anda saat peran ini aktif. Ada saran nama, berdasarkan informasi akun dan peran, tetapi Anda dapat mengubahnya menjadi apa pun yang berarti bagi Anda.
   - Untuk Warna tampilan pilih warna untuk menyorot nama tampilan.

   Nama dan warna dapat membantu mengingatkan Anda saat peran ini aktif, yang mengubah izin Anda. Misalnya, untuk peran yang memberi Anda akses ke lingkungan pengujian, Anda dapat menentukan Nama tampilan dari **`Test`** dan pilih Warna hijau. Untuk peran yang memberi Anda akses ke produksi, Anda dapat menentukan Nama tampilan dari **`Production`** dan pilih merah sebagai Warna.
7. Pilih Ganti Peran. Nama tampilan dan warna menggantikan nama pengguna Anda pada bilah navigasi, dan Anda dapat mulai menggunakan izin yang diberikan peran tersebut kepada Anda.
8. Setelah Anda menyelesaikan tugas-tugas yang memerlukan IAM peran, Anda dapat beralih kembali ke sesi awal Anda. Ini akan menghapus izin tambahan yang diberikan oleh peran dan mengembalikan Anda ke izin standar Anda.

a. Di IAM konsol, pilih nama Tampilan peran Anda di bilah navigasi di kanan atas.

b. Pilih Beralih kembali.

   Misalnya, anggap Anda masuk ke nomor akun `123456789012` menggunakan nama pengguna `RichardRoe`. Setelah menggunakan peran `admin-role`, Anda ingin berhenti menggunakan peran dan kembali ke izin awal. Untuk berhenti menggunakan peran, pilih peran admin @ 123456789012, lalu pilih Beralih kembali.



> **Tip**
>
> Beberapa peran terakhir yang Anda gunakan muncul pada menu. Lain kali Anda ingin beralih ke salah satu peran itu, Anda cukup memilih peran yang Anda inginkan. Anda hanya perlu mengetikkan akun dan informasi peran secara manual jika peran tidak ditampilkan di menu.

Sumber daya tambahan

- Berikan izin pengguna untuk beralih peran
- Berikan izin pengguna untuk meneruskan peran ke layanan AWS
- Membuat peran untuk mendelegasikan izin ke pengguna IAM
- Membuat peran untuk mendelegasikan izin ke layanan AWS
- Memecahkan masalah peran IAM

Beralih ke IAM peran (AWS CLI)

Peran menentukan sekumpulan izin yang dapat Anda gunakan untuk mengakses AWS sumber daya yang Anda butuhkan. Dalam hal ini, ini mirip dengan pengguna di AWS Identity and Access Management(IAM). Saat Anda masuk sebagai pengguna, Anda mendapatkan serangkaian izin tertentu. Namun, Anda tidak masuk ke sebuah peran, tetapi setelah masuk sebagai pengguna, Anda dapat beralih ke sebuah peran. Sementara waktu ini mengesampingkan izin pengguna awal Anda dan justru memberikan izin yang ditetapkan untuk peran. Peran dapat di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi selengkapnya tentang peran, manfaatnya, dan cara membuat dan mengonfigurasinya, lihatIAMperan, danIAMpenciptaan peran. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk mengasumsikan peran, lihat Metode untuk mengambil peran.

> **Important**
>
> Izin IAM pengguna Anda dan peran apa pun yang Anda asumsikan tidak kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda mengasumsikan suatu peran, sementara waktu Anda meninggalkan izin pengguna dan bekerja dengan izin yang ditetapkan ke peran tersebut. Saat Anda keluar dari peran, izin pengguna asli Anda dipulihkan secara otomatis.

Anda dapat menggunakan peran untuk menjalankan AWS CLI perintah saat Anda masuk sebagai IAM pengguna. Anda juga dapat menggunakan peran untuk menjalankan AWS CLI perintah saat Anda masuk sebagai pengguna yang diautentikasi secara eksternal (SAMLatau OIDC) yang sudah menggunakan peran. Selain itu, Anda dapat menggunakan peran untuk menjalankan AWS CLI perintah dari dalam EC2 instance Amazon yang dilampirkan ke peran melalui profil instance-nya. Anda tidak dapat mengambil peran ketika Anda masuk sebagai Pengguna root akun AWS.

Rantai peran — Anda juga dapat menggunakan rantai peran, yang menggunakan izin dari peran untuk mengakses peran kedua.

Secara default, sesi peran Anda berlangsung selama satu jam. Ketika Anda mengambil peran ini menggunakan `assume-role*` CLI operasi, Anda dapat menentukan nilai untuk `duration-seconds` parameter. Nilai ini dapat berkisar dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Jika Anda beralih peran di konsol, durasi sesi Anda dibatasi hingga maksimal satu jam. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Saat Anda menggunakan rantai peran, kredensial baru Anda dibatasi hingga durasi maksimum satu jam. Jika Anda kemudian menggunakan parameter `duration-seconds` untuk memberikan nilai lebih dari satu jam, operasi gagal.

Skenario contoh: Beralih ke peran produksi

Bayangkan Anda adalah IAM pengguna untuk bekerja di lingkungan pengembangan. Dalam skenario ini, Anda kadang-kadang perlu bekerja dengan lingkungan produksi di baris perintah dengan AWS CLI. Anda sudah memiliki set kredensi kunci akses yang tersedia untuk Anda. Ini bisa menjadi access key pair yang ditetapkan untuk IAM pengguna standar Anda. Atau, jika Anda masuk sebagai pengguna gabungan, itu dapat menjadi pasangan access key untuk peran yang awalnya ditetapkan kepada Anda. Jika izin Anda saat ini memberi Anda kemampuan untuk mengambil IAM peran tertentu, maka Anda dapat mengidentifikasi peran tersebut dalam “profil” di AWS CLI file konfigurasi. Perintah itu kemudian dijalankan dengan izin dari IAM peran yang ditentukan, bukan identitas asli. Perhatikan bahwa ketika Anda menentukan profil itu di AWS CLI perintah, Anda menggunakan peran baru. Dalam situasi ini, Anda tidak dapat menggunakan izin awal dalam akun pengembangan pada saat bersamaan. Alasannya adalah bahwa hanya satu rangkaian izin yang dapat berlaku pada satu waktu.

> **Note**
>
> Untuk tujuan keamanan, administrator dapat meninjau AWS CloudTrail log untuk mempelajari siapa yang melakukan tindakan di AWS. Administrator Anda mungkin mengharuskan Anda menentukan identitas sumber atau nama sesi peran saat Anda mengambil peran tersebut. Untuk informasi selengkapnya, silakan lihat `sts:SourceIdentity` dan `sts:RoleSessionName`.

Untuk beralih ke peran produksi (AWS CLI)

1. Jika Anda belum pernah menggunakan AWS CLI, maka Anda harus terlebih dahulu mengkonfigurasi CLI profil default Anda. Buka prompt perintah dan atur AWS CLI instalasi untuk menggunakan kunci akses dari IAM pengguna Anda atau dari peran federasi Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS Command Line Interfacedi AWS Command Line Interface Panduan Pengguna.

   Jalankan perintah aws configure sebagai berikut:

   ```
   aws configure
   ```

   Saat diminta, berikan informasi berikut:

   ```
   AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   Default region name [None]: us-east-2
   Default output format [None]: json
   ```

2. Buat profil baru untuk peran dalam `.aws/config` file dalam Unix atau Linux, atau `C:\Users\USERNAME\.aws\config` file dalam Windows. Contoh berikut membuat sebuah profil bernama `prodaccess` yang beralih ke peran *`ProductionAccessRole`* dalam akun `123456789012`. Anda mendapatkan peran ARN dari administrator akun yang membuat peran tersebut. Ketika profil ini dipanggil, AWS CLI menggunakan kredensi `source_profile` untuk meminta kredensional untuk peran tersebut. Oleh karena itu, identitas direferensikan karena `source_profile` harus memiliki izin `sts:AssumeRole` untuk peran yang ditentukan dalam `role_arn`.

   ```
   [profile prodaccess]
       role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
       source_profile = default
   ```

3. Setelah Anda membuat profil baru, apa saja AWS CLI perintah yang menentukan parameter `--profile prodaccess` berjalan di bawah izin yang dilampirkan ke IAM peran `ProductionAccessRole` bukan pengguna default.

   ```
   aws iam list-users --profile prodaccess
   ```

Perintah ini berfungsi jika izin yang ditetapkan untuk `ProductionAccessRole` mengaktifkan daftar pengguna di saat ini AWS akun.

4. Untuk kembali ke izin yang diberikan oleh kredensial asli Anda, jalankan perintah tanpa parameter `--profile`. Bagian AWS CLI kembali menggunakan kredensi di profil default Anda, yang Anda konfigurasikan. Step 1

Untuk informasi selengkapnya, lihat Mengasumsikan Peran dalam AWS Command Line Interface Panduan Pengguna.

Skenario contoh: Menginzinkan peran profil instans untuk beralih ke peran dalam akun lain

Bayangkan Anda menggunakan dua Akun AWS, dan Anda ingin mengizinkan aplikasi yang berjalan pada EC2 instance Amazon untuk dijalankan AWS CLIperintah di kedua akun. Asumsikan bahwa EC2 instance ada di akun`111111111111`. Instance tersebut menyertakan peran profil `abcd` instance yang memungkinkan aplikasi melakukan tugas Amazon S3 hanya-baca di bucket dalam akun `amzn-s3-demo-bucket1` yang sama. `111111111111` Namun, aplikasi tersebut juga harus diizinkan untuk mengambil peran lintas akun `efgh` untuk melakukan tugas di akun `222222222222`. Untuk melakukan ini, peran profil `abcd` EC2 instance harus memiliki kebijakan izin berikut:

Kebijakan izin peran akun 1111111111 ***abcd***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
```

```
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Anggap bahwa `efgh` peran lintas akun memungkinkan tugas Amazon S3 hanya baca di bucket `amzn-s3-demo-bucket2` dengan akun `222222222222` yang sama. Untuk melakukannya, peran lintas akun `efgh` harus memiliki kebijakan izin berikut:

Kebijakan izin peran akun 2222222222 ***efgh***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
```

```
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

Peran `efgh` harus memungkinkan peran profil instans `abcd` untuk mengasumsikannya. Untuk melakukannya, peran `efgh` harus memiliki kebijakan kepercayaan berikut:

Akun 222222222222 kebijakan kepercayaan peran ***`efgh`***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Untuk kemudian berlari AWS CLI perintah di akun222222222222, Anda harus memperbarui file CLI konfigurasi. Identifikasi `efgh` peran sebagai “profil” dan peran profil `abcd` EC2 instance sebagai “sumber kredensi” di AWS CLI file konfigurasi. Kemudian CLI perintah Anda dijalankan dengan izin `efgh` peran, bukan `abcd` peran asli.

> **Note**
>
> Untuk tujuan keamanan, Anda dapat menggunakan AWS CloudTrail untuk mengaudit penggunaan peran dalam akun. Untuk membedakan antara sesi peran ketika peran digunakan oleh prinsipal yang berbeda dalam CloudTrail log, Anda dapat menggunakan nama sesi peran. Saat AWS CLI mengasumsikan peran atas nama pengguna seperti yang dijelaskan dalam topik ini, nama sesi peran secara otomatis dibuat sebagai`AWS-CLI-session-`*`nnnnnnnn`*. Di sini *`nnnnnnnn`* adalah bilangan bulat yang mewakili waktu dalam waktu epoch Unix (jumlah detik sejak tengah malam UTC pada tanggal 1 Januari 1970).

Untuk informasi selengkapnya, lihat Referensi CloudTrail Acara di AWS CloudTrail Panduan Pengguna.

Untuk mengizinkan peran profil EC2 instance beralih ke peran lintas akun (AWS CLI)

1. Anda tidak perlu mengkonfigurasi CLI profil default. Sebagai gantinya, Anda dapat memuat kredensional dari metadata profil EC2 instance. Buat profil baru untuk peran dalam file `.aws/config`. Contoh berikut membuat profil `instancecrossaccount` yang beralih ke peran *`efgh`* dalam akun `222222222222`. Ketika profil ini dipanggil, AWS CLI menggunakan kredensi metadata profil EC2 instance untuk meminta kredensi peran tersebut. Karena itu, peran profil EC2 instance harus memiliki `sts:AssumeRole` izin untuk peran yang ditentukan dalam. `role_arn`

   ```
   [profile instancecrossaccount]
   role_arn = arn:aws:iam::222222222222:role/efgh
   credential_source = Ec2InstanceMetadata
   ```

2. Setelah Anda membuat profil baru, apa saja AWS CLI perintah yang menentukan parameter `--profile instancecrossaccount` berjalan di bawah izin yang dilampirkan ke `efgh` peran dalam akun. `222222222222`

   ```
   aws s3 ls amzn-s3-demo-bucket2 --profile instancecrossaccount
   ```

   Perintah ini berfungsi jika izin yang ditetapkan untuk `efgh` peran memungkinkan daftar pengguna di saat ini Akun AWS.

3. Untuk kembali ke izin profil EC2 instans asli di akun`111111111111`, jalankan CLI perintah tanpa `--profile` parameter.

Untuk informasi selengkapnya, lihat Mengasumsikan Peran dalam AWS Command Line Interface Panduan Pengguna.

Beralih ke IAM peran (Alat untuk Windows PowerShell)

Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS yang Anda perlukan. Dalam pengertian itu, ini mirip dengan pengguna di AWS Identity and Access Management (IAM). Saat Anda masuk sebagai pengguna, Anda mendapatkan serangkaian izin tertentu. Namun, Anda tidak masuk ke sebuah peran, tetapi setelah masuk, Anda dapat beralih

ke sebuah peran. Sementara waktu ini mengesampingkan izin pengguna awal Anda dan justru memberikan izin yang ditetapkan untuk peran. Perannya bisa di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat IAMperan, dan IAMpenciptaan peran.

> **Important**
>
> Izin IAM pengguna Anda dan peran apa pun yang Anda alihkan tidak bersifat kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda beralih ke suatu peran, sementara Anda meninggalkan izin pengguna dan bekerja dengan izin yang diberikan ke peran tersebut. Saat Anda keluar dari peran, izin pengguna asli Anda dipulihkan secara otomatis.

Bagian ini menjelaskan cara berganti peran ketika Anda bekerja di baris perintah dengan AWS Tools for Windows PowerShell.

Bayangkan Anda memiliki akun di lingkungan pengembangan dan Anda kadang-kadang perlu bekerja dengan lingkungan produksi di baris perintah menggunakan Alat untuk Windows PowerShell. Anda sudah memiliki satu set kredensial access key yang tersedia untuk Anda. Ini bisa berupa access key pair yang ditetapkan untuk IAM pengguna standar Anda. Atau, jika Anda masuk sebagai pengguna gabungan, itu dapat menjadi pasangan access key untuk peran yang awalnya ditetapkan kepada Anda. Anda dapat menggunakan kredenal ini untuk menjalankan `Use-STSRole` cmdlet yang meneruskan peran baru sebagai parameter. ARN Perintah tersebut mengembalikan kredensial keamanansementara untuk peran yang diminta. Anda kemudian dapat menggunakan kredenal tersebut dalam PowerShell perintah berikutnya dengan izin peran untuk mengakses sumber daya dalam produksi. Saat menggunakan peran tersebut, Anda tidak dapat menggunakan izin pengguna dalam akun Pengembangan karena hanya satu rangkaian izin yang berlaku pada satu waktu.

> **Note**
>
> Untuk tujuan keamanan, administrator dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan. AWS Administrator Anda mungkin akan meminta Anda menentukan identitas sumber atau nama sesi peran ketika Anda mengambil peran tersebut. Untuk informasi selengkapnya, silakan lihat `sts:SourceIdentity` dan `sts:RoleSessionName`.

Perhatikan bahwa semua kunci akses dan token hanyalah contoh dan tidak dapat digunakan seperti yang ditunjukkan. Ganti dengan nilai yang sesuai dari lingkungan langsung Anda.

Untuk beralih ke peran (Alat untuk Windows PowerShell)

1. Buka prompt PowerShell perintah dan konfigurasikan profil default untuk menggunakan kunci akses dari IAM pengguna Anda saat ini atau dari peran federasi Anda. Jika sebelumnya Anda telah menggunakan Alat untuk Windows PowerShell, maka ini kemungkinan sudah dilakukan. Perhatikan bahwa Anda dapat beralih peran hanya jika Anda masuk sebagai IAM pengguna, bukan Pengguna root akun AWS.

   ```
   PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
   SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
   PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
   ```

   Untuk informasi selengkapnya, lihat Menggunakan AWS Kredensial di AWS Tools for Windows PowerShell Panduan Pengguna.

2. Untuk mendapatkan kredensial peran baru, jalankan perintah berikut untuk beralih ke peran *RoleName* dalam akun 123456789012. Anda mendapatkan peran ARN dari administrator akun yang membuat peran tersebut. Perintah ini mengharuskan Anda untuk juga memberikan nama sesi. Anda dapat memilih teks apa pun untuk itu. Perintah berikut meminta kredensial dan kemudian menangkap objek properti `Credentials` dari objek hasil yang dikembalikan dan menyimpannya dalam variabel `$Creds`.

   ```
   PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
   RoleSessionName "MyRoleSessionName").Credentials
   ```

   `$Creds` adalah objek yang sekarang berisi elemen `AccessKeyId`, `SecretAccessKey`, dan `SessionToken` yang Anda butuhkan dalam langkah-langkah berikut ini. Perintah sampel berikut ini menggambarkan nilai-nilai umum:

   ```
   PS C:\> $Creds.AccessKeyId
   AKIAIOSFODNN7EXAMPLE

   PS C:\> $Creds.SecretAccessKey
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

   PS C:\> $Creds.SessionToken
   ```

```
AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. Untuk menggunakan kredensial ini untuk perintah berikutnya, sertakan mereka dengan parameter `-Credential`. Misalnya, perintah berikut menggunakan kredensial dari peran dan bekerja hanya jika peran tersebut diberik izin`iam:ListRoles` dan karenanya dapat menjalankan `Get-IAMRoles` cmdlet:

```
PS C:\> get-iamroles -Credential $Creds
```

4. Untuk kembali ke kredensi asli Anda, cukup berhenti menggunakan `-Credentials $Creds` parameter dan izinkan PowerShell untuk kembali ke kredensi yang disimpan di profil default.

Beralih ke IAM peran (AWS API)

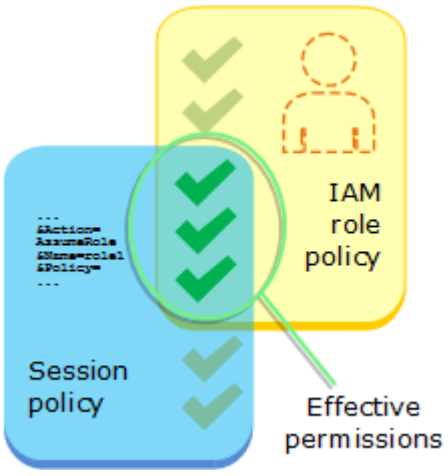
Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS . Dalam hal ini, ini mirip dengan IAMpengguna. Seorang kepala sekolah (orang atau aplikasi) mengambil peran untuk menerima izin sementara untuk melaksanakan tugas yang diperlukan dan berinteraksi dengan AWS sumber daya. Perannya bisa di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat IAMperan, dan IAMpenciptaan peran. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk mengasumsikan peran, lihat Metode untuk mengambil peran.

> **Important**
>
> Izin IAM pengguna Anda dan peran apa pun yang Anda asumsikan tidak kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda mengasumsikan suatu peran, sementara waktu Anda meninggalkan izin pengguna dan bekerja dengan izin yang ditetapkan

ke peran tersebut. Saat Anda keluar dari peran tersebut, izin asli Anda akan dipulihkan secara otomatis.

Untuk mengambil peran, aplikasi memanggil AWS STS `AssumeRole`APIoperasi dan meneruskan peran ARN yang akan digunakan. Operasi membuat sesi baru dengan kredensial sementara. Sesi ini memiliki izin yang sama dengan kebijakan berbasis identitas untuk peran tersebut.

Saat Anda menelepon `AssumeRole`, Anda secara opsional dapat meneruskan kebijakan sesi inline atau terkelola. Kebijakan sesi adalah kebijakan yang Anda teruskan sebagai parameter saat Anda secara membuat sesi sementara terprogram untuk peran atau pengguna federasi. Anda dapat meneruskan satu dokumen kebijakan sesi JSON inline menggunakan `Policy` parameter. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas entitas dan kebijakan sesi. Kebijakan sesi berguna saat Anda perlu memberikan kredensial sementara kepada orang lain. Mereka dapat menggunakan kredensi sementara peran dalam AWS API panggilan berikutnya untuk mengakses sumber daya di akun yang memiliki peran tersebut. Anda tidak dapat menggunakan kebijakan sesi untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas. Untuk mempelajari selengkapnya tentang cara AWS menentukan izin efektif suatu peran, lihatLogika evaluasi kebijakan.



Anda dapat menelepon `AssumeRole` saat masuk sebagai IAM pengguna, atau sebagai pengguna yang diautentikasi secara eksternal (SAMLatau OIDC) yang sudah menggunakan peran. Anda juga dapat menggunakan rantai peran, yang menggunakan peran untuk mengasumsikan peran kedua. Anda tidak dapat mengambil peran saat masuk sebagai Pengguna root akun AWS.

Secara default, sesi peran Anda berlangsung selama satu jam. Saat Anda mengasumsikan peran ini menggunakan AWS STS `AssumeRole*`APIoperasi, Anda dapat menentukan nilai untuk

`DurationSeconds` parameter tersebut. Nilai ini dapat berkisar dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Jika Anda menggunakan rantai peran, sesi Anda akan dibatasi hingga maksimum satu jam. Jika Anda kemudian menggunakan parameter `DurationSeconds` untuk memberikan nilai lebih dari satu jam, operasi gagal.

> **Note**
>
> Untuk tujuan keamanan, administrator dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan. AWS Administrator Anda mungkin akan meminta Anda menentukan identitas sumber atau nama sesi peran ketika Anda mengambil peran tersebut. Untuk informasi selengkapnya, silakan lihat `sts:SourceIdentity` dan `sts:RoleSessionName`.

Contoh kode berikut menunjukkan cara membuat pengguna dan mengambil peran.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan IAM pengguna untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat pengguna tanpa izin.
- Buat peran yang memberikan izin untuk mencantumkan bucket Amazon S3 untuk akun tersebut.
- Tambahkan kebijakan agar pengguna dapat mengambil peran tersebut.
- Asumsikan peran dan daftar bucket S3 menggunakan kredensyal sementara, lalu bersihkan sumber daya.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
```

```
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
```

```
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
```

```
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
```

```
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
 { UserName = userName });
        return response.User;
```

```
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
```

```
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
```

```
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
```

```
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
```

```
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
```

```
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");
```

```
        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);
```

```
        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();
```

```
        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
```

```
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
```

```
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
```

```
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
```

```
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
```

```
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for .NET APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
```

```
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
```

```
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
```

```
 fi

 local assume_role_policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"sts:AssumeRole\",
                  \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
```

```
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

IAMFungsi yang digunakan dalam skenario ini.

```
###############################################################################
# function iam_user_exists
#
```

```
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
```

```
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
```

```
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_user_access_key"
  echo "Creates an AWS Identity and Access Management (IAM) key pair."
  echo "  -u user_name   The name of the IAM user."
  echo "  [-f file_name]   Optional file name for the access key output."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    f) file_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
```

```
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }
```

```
# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
  return 1
fi
```

```
  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
```

```
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
```

```
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
```

```
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi

iecho "delete-access-key response:$response"
iecho
```

```
  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Untuk API detailnya, lihat topik berikut di Referensi AWS CLI Perintah.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;
```

```
    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }

        user = outcome.GetResult().GetUser();
    }
```

```
    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
```

```
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
```

```
                        listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
```

```
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
```

```
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }
```

```
    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for C++ APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy

- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Jalankan skenario interaktif di penggugah/prompt perintah.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
```

```
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
```

```
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
```

```
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
```

```
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
```

```
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
```

```
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
```

```
 }
 for _, policy := range policies {
  err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
*policy.PolicyArn)
  if err != nil {
   panic(err)
  }
  err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
  if err != nil {
   panic(err)
  }
  log.Printf("Detached policy %v from role %v and deleted the policy.\n",
   *policy.PolicyName, *role.RoleName)
 }
 err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Deleted role %v.\n", *role.RoleName)

 userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 for _, userPol := range userPols {
  err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
 }
 keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 for _, key := range keys {
  err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
*key.AccessKeyId)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
*user.UserName)
 }
```

```
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
  log.Println(strings.Repeat("-", 88))
 }

}
```

Tentukan struct yang membungkus tindakan akun.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

```
// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Tentukan struct yang membungkus tindakan kebijakan.

```
// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}
```

```
// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
```

```
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan peran.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
```

```
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
```

```
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
```

```
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
```

```
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan pengguna.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
```

```
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
```

```
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

```
// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
```

```
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Go APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM pengguna.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
```

```
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

   public static String userArn;

   public static void main(String[] args) throws Exception {

       final String usage = """

               Usage:
                   <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

               Where:
                   username - The name of the IAM user to create.\s
                   policyName - The name of the policy to create.\s
                   roleName - The name of the role to create.\s
                   roleSessionName - The name of the session required for the
assumeRole operation.\s
                   bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
               """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
       String policyName = args[1];
       String roleName = args[2];
       String roleSessionName = args[3];
       String bucketName = args[4];

       Region region = Region.AWS_GLOBAL;
       IamClient iam = IamClient.builder()
               .region(region)
               .build();

       System.out.println(DASHES);
       System.out.println("Welcome to the AWS IAM example scenario.");
       System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();
```

```
            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
```

```
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }
```

```
            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();
```

```
            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);
```

```
            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
```

```
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Java 2.x APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
```

```
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
```

```
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
```

```
              Principal: {
                // Allow the previously created user to assume this role.
                AWS: User.Arn,
              },
              Action: "sts:AssumeRole",
            },
          ],
        }),
        RoleName: roleName,
      }),
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
```

```
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
  listBuckets(s3Client),
);
```

```
  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }
```

```
  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for JavaScript APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM pengguna.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>
```

```
    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}
```

```
suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue: String =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}"

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
```

```
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
```

```
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient =
        StsClient {
            region = "us-east-1"
        }

    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials =
        StaticCredentialsProvider {
```

```
            accessKeyId = key
            secretAccessKey = secKey
            sessionToken = secToken
        }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 =
        S3Client {
            credentialsProvider = staticCredentials
            region = "us-east-1"
        }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
```

```
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk API referensi Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
```

```
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
```

```
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for PHP APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
```

```
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)
```

```
    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
```

```
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
"
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)

    return user, user_key, role
```

```
def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
```

```
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
```

```
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
```

```
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
 user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
```

```
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 'sts:AssumeRole',
        Resource: role.arn
      }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
```

```
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an Amazon S3 resource with specified credentials. This is separated
into a
  # factory function so that it can be mocked for unit testing.
  #
  # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
  def create_s3_resource(credentials)
    Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
  end

  # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
  # Because the resource uses credentials with limited access, it may not be able
to
  # list the S3 buckets.
  #
  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def list_buckets(s3_resource)
    count = 10
    s3_resource.buckets.each do |bucket|
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == 'AccessDenied'
      puts('Attempt to list buckets with no permissions: AccessDenied.')
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
```

```
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
```

```
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Ruby APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
```

```
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;
```

```
    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
```

```
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
```

```
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
```

```
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk API referensi Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2

Aplikasi yang berjalan pada EC2 instans Amazon harus menyertakan AWS kredensi di AWS APIpermintaan. Anda dapat memiliki toko pengembang Anda AWS kredensil langsung dalam EC2 instans Amazon dan memungkinkan aplikasi dalam instance tersebut untuk menggunakan kredensil tersebut. Tetapi pengembang kemudian harus mengelola kredensil dan memastikan bahwa mereka

meneruskan kredensil dengan aman ke setiap EC2 instance dan memperbarui setiap instans Amazon ketika saatnya memperbarui kredensil. Banyak pekerjaan tambahan.

Sebagai gantinya, Anda dapat dan harus menggunakan IAM peran untuk mengelola kredensional sementara untuk aplikasi yang berjalan di instans AmazonEC2. Saat Anda menggunakan peran, Anda tidak perlu mendistribusikan kredensil jangka panjang (seperti kredenal masuk atau kunci akses) ke instans Amazon. EC2 Sebagai gantinya, peran menyediakan izin sementara yang dapat digunakan aplikasi saat mereka melakukan panggilan ke orang lain AWS sumber daya. Saat meluncurkan EC2 instans Amazon, Anda menentukan IAM peran yang akan diasosiasikan dengan instance tersebut. Aplikasi yang berjalan pada instance kemudian dapat menggunakan kredenal sementara yang disediakan peran untuk menandatangani permintaan. API

Menggunakan peran untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon memerlukan sedikit konfigurasi tambahan. Aplikasi yang berjalan pada EC2 instance Amazon diabstraksikan dari AWS dengan sistem operasi tervirtualisasi. Karena pemisahan ekstra ini, Anda memerlukan langkah tambahan untuk menetapkan AWS peran dan izin terkait ke EC2 instans Amazon dan membuatnya tersedia untuk aplikasinya. Langkah ekstra ini adalah pembuatan profil instance yang dilampirkan ke instance. Profil instans berisi peran dan dapat memberikan kredensial sementara peran ke aplikasi yang berjalan pada instans. Kredensi sementara tersebut kemudian dapat digunakan dalam API panggilan aplikasi untuk mengakses sumber daya dan untuk membatasi akses hanya ke sumber daya yang ditentukan oleh peran tersebut.

> **Note**
>
> Hanya satu peran yang dapat ditetapkan ke EC2 instans Amazon pada satu waktu, dan semua aplikasi pada instance berbagi peran dan izin yang sama. Saat memanfaatkan Amazon ECS untuk mengelola EC2 instans Amazon, Anda dapat menetapkan peran ke ECS tugas Amazon yang dapat dibedakan dari peran EC2 instans Amazon yang dijalankannya. Menetapkan setiap tugas peran selaras dengan prinsip akses yang paling tidak memiliki hak istimewa dan memungkinkan kontrol terperinci yang lebih besar atas tindakan dan sumber daya.
> Untuk informasi selengkapnya, lihat Menggunakan IAM peran dengan ECS tugas Amazon di Panduan Praktik Terbaik Amazon Elastic Container Service.

Penggunaan peran dengan cara ini memiliki beberapa keuntungan. Karena kredensi peran bersifat sementara dan diperbarui secara otomatis, Anda tidak perlu mengelola kredensil, dan Anda tidak perlu khawatir tentang risiko keamanan jangka panjang. Selain itu, jika Anda menggunakan satu

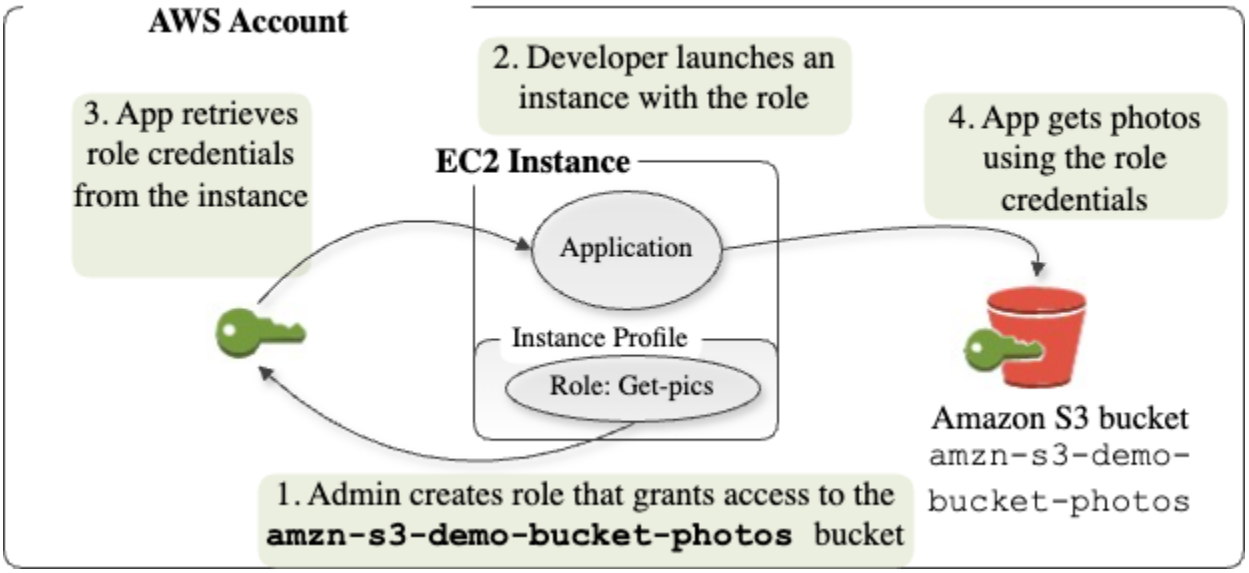
peran untuk beberapa instance, Anda dapat membuat perubahan pada satu peran tersebut dan perubahan tersebut menyebar secara otomatis ke semua instance.

> **Note**
>
> Meskipun peran biasanya ditetapkan ke EC2 instans Amazon saat Anda meluncurkannya, peran juga dapat dilampirkan ke EC2 instans Amazon yang sedang berjalan. Untuk mempelajari cara melampirkan peran ke instance yang sedang berjalan, lihat IAMPeran untuk Amazon EC2.

Topik

Bagaimana cara kerja peran untuk EC2 instans Amazon?

Pada gambar berikut, pengembang menjalankan aplikasi pada EC2 instance Amazon yang memerlukan akses ke bucket S3 bernama`amzn-s3-demo-bucket-photos`. Administrator membuat peran `Get-pics` layanan dan melampirkan peran tersebut ke EC2 instans Amazon. Peran ini mencakup kebijakan izin yang memberikan akses hanya-baca ke bucket S3 tertentu. Ini juga mencakup kebijakan kepercayaan yang memungkinkan EC2 instans Amazon untuk mengambil peran dan mengambil kredensi sementara. Saat aplikasi berjalan pada instans, itu menggunakan kredensial sementara peran untuk mengakses bucket foto. Administrator tidak perlu memberikan izin kepada developer untuk mengakses bucket foto, dan developer tidak perlu membagikan atau mengelola kredensial.



1. Administrator menggunakan IAM untuk membuat **`Get-pics`** peran. Dalam kebijakan kepercayaan peran, administrator menetapkan bahwa hanya EC2 instans Amazon yang dapat mengambil peran tersebut. Dalam kebijakan izin peran, administrator menentukan izin hanya-baca untuk bucket `amzn-s3-demo-bucket-photos`.
2. Pengembang meluncurkan EC2 instance Amazon dan menetapkan `Get-pics` peran tersebut ke instance tersebut.

   > **Note**
   >
   > Jika Anda menggunakan IAM konsol, profil instance dikelola untuk Anda dan sebagian besar transparan bagi Anda. Namun, jika Anda menggunakan AWS CLI atau API untuk membuat dan mengelola peran dan EC2 instans Amazon, maka Anda harus membuat profil instance dan menetapkan peran sebagai langkah terpisah. Kemudian, saat Anda meluncurkan instans, Anda harus menentukan nama profil instans dan bukan nama peran.

3. Saat aplikasi berjalan, aplikasi memperoleh kredensil keamanan sementara dari metadata EC2 instans Amazon, seperti yang dijelaskan dalam Mengambil Kredensial Keamanan dari Metadata Instance. Ini adalah kredensial keamanan sementara yang mewakili peran tersebut dan berlaku untuk periode waktu yang terbatas.

   Dengan beberapa AWS SDKs, pengembang dapat menggunakan penyedia yang mengelola kredensi keamanan sementara secara transparan. (Dokumentasi untuk individu AWS SDKsmenjelaskan fitur yang didukung oleh itu SDK untuk mengelola kredensional.)

Atau, aplikasi bisa mendapatkan kredensi sementara langsung dari metadata instance dari instans Amazon. EC2 Kredensial dan nilai terkait tersedia dari kategori `iam/security-credentials/`*`role-name`* (dalam hal ini, `iam/security-credentials/Get-pics`) metadata. Jika aplikasi mendapatkan kredensial dari metadata instans, itu dapat menyimpan kredensial tersebut.

4. Dengan menggunakan kredensial sementara yang diambil, aplikasi mengakses bucket foto. Karena kebijakan terlampir pada peran **`Get-pics`**, aplikasi memiliki izin hanya-baca.

   Kredensi keamanan sementara yang tersedia pada instans secara otomatis diperbarui sebelum kedaluwarsa sehingga set yang valid selalu tersedia. Aplikasi hanya perlu memastikan bahwa itu mendapatkan serangkaian kredensial baru dari metadata instans sebelum metadata yang sedang digunakan kedaluwarsa. Hal ini dimungkinkan untuk menggunakan AWS SDKuntuk mengelola kredensil sehingga aplikasi tidak perlu menyertakan logika tambahan untuk menyegarkan kredensil. Misalnya, membuat instan klien dengan Penyedia Kredensial Profil Instans. Namun, jika aplikasi mendapatkan kredensial keamanan sementara dari metadata instans dan sudah menyimpannya, itu akan mendapatkan rangkaian kredensial yang diperbarui setiap jam, atau setidaknya 15 menit sebelum rangkaian yang sedang digunakan kedaluwarsa. Waktu kedaluwarsa termasuk dalam informasi yang dikembalikan dalam kategori. `iam/security-credentials/`*`role-name`*

Izin diperlukan untuk menggunakan peran dengan Amazon EC2

Untuk meluncurkan instance dengan peran, pengembang harus memiliki izin untuk meluncurkan EC2 instans Amazon dan izin untuk meneruskan IAM peran.

Kebijakan contoh berikut memungkinkan pengguna untuk menggunakan AWS Management Console untuk meluncurkan instance dengan peran. Kebijakan ini mencakup wildcard (`*`) untuk memungkinkan pengguna meneruskan peran apa pun dan melakukan EC2 tindakan Amazon yang terdaftar. `ListInstanceProfiles`Tindakan ini memungkinkan pengguna untuk melihat semua peran yang tersedia di Akun AWS.

Example Contoh kebijakan yang memberikan izin kepada pengguna untuk menggunakan EC2 konsol Amazon untuk meluncurkan instance dengan peran apa pun

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

Membatasi peran mana yang dapat diteruskan ke EC2 instans Amazon (menggunakan) PassRole

Anda dapat menggunakan `PassRole` izin untuk membatasi peran mana yang dapat diteruskan pengguna ke EC2 instans Amazon saat pengguna meluncurkan instance. Ini membantu mencegah pengguna menjalankan aplikasi yang memiliki izin lebih dari yang diberikan kepada pengguna—yaitu, agar dapat memperoleh hak istimewa yang ditingkatkan. Misalnya, bayangkan bahwa pengguna Alice memiliki izin hanya untuk meluncurkan EC2 instans Amazon dan bekerja dengan bucket Amazon S3, tetapi peran yang dia berikan ke EC2 instance Amazon memiliki izin untuk bekerja dengan dan Amazon DynamoDB. IAM Dalam hal ini, Alice mungkin dapat meluncurkan instance, masuk ke dalamnya, mendapatkan kredensil keamanan sementara, dan kemudian melakukan atau tindakan IAM DynamoDB yang tidak dia otorisasi.

Untuk membatasi peran mana yang dapat diteruskan pengguna ke EC2 instans Amazon, Anda membuat kebijakan yang mengizinkan `PassRole` tindakan tersebut. Anda kemudian melampirkan kebijakan ke pengguna (atau ke IAM grup yang menjadi milik pengguna) yang akan meluncurkan EC2 instans Amazon. Di `Resource` elemen kebijakan, Anda mencantumkan peran atau peran yang diizinkan diteruskan pengguna ke EC2 instans Amazon. Saat pengguna meluncurkan instance

dan mengaitkan peran dengannya, Amazon EC2 memeriksa apakah pengguna diizinkan untuk melewati peran itu. Tentu saja, Anda juga harus memastikan bahwa peran yang dapat diteruskan oleh pengguna tidak mencakup lebih banyak izin dari yang seharusnya dimiliki oleh pengguna.

> **Note**
>
> `PassRole`bukan API tindakan dengan cara yang sama seperti itu `RunInstances` atau `ListInstanceProfiles` apa adanya. Sebaliknya, itu adalah izin yang AWS memeriksa setiap kali peran ARN diteruskan sebagai parameter ke API (atau konsol melakukan ini atas nama pengguna). Itu membantu administrator untuk mengontrol peran mana yang dapat diteruskan oleh pengguna yang mana. Dalam hal ini, ini memastikan bahwa pengguna diizinkan untuk melampirkan peran tertentu ke EC2 instance Amazon.

Example Contoh kebijakan yang memberikan izin kepada pengguna untuk meluncurkan EC2 instans Amazon dengan peran tertentu

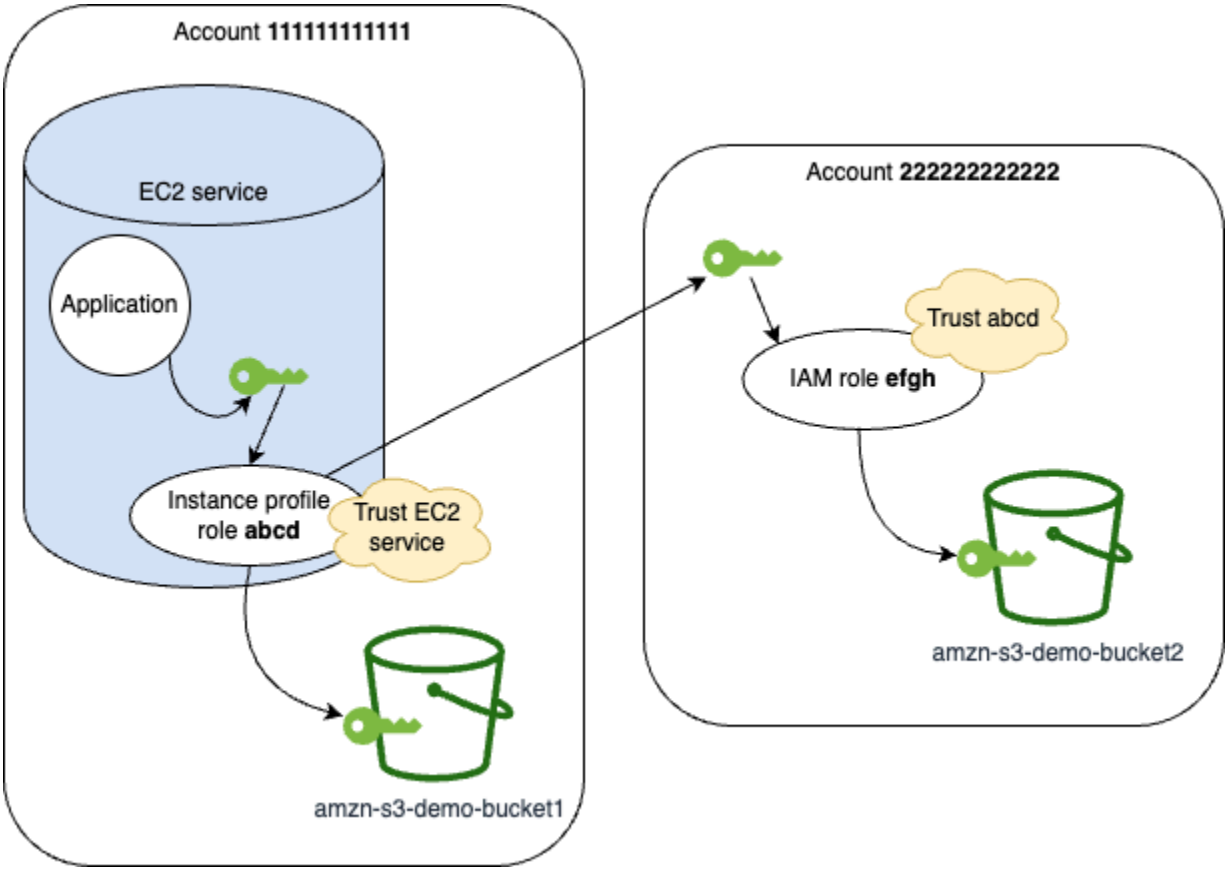
Kebijakan contoh berikut memungkinkan pengguna menggunakan Amazon EC2 API untuk meluncurkan instance dengan peran. `Resource`Elemen menentukan Amazon Resource Name (ARN) peran. Dengan menentukanARN, kebijakan memberi pengguna izin untuk hanya meneruskan peran. `Get-pics` Jika pengguna mencoba menentukan peran yang berbeda saat meluncurkan suatu instans, tindakan tersebut gagal. Pengguna memang memiliki izin untuk menjalankan setiap instans, terlepas dari apakah mereka meneruskan suatu peran.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id:role/Get-pics"
    }
  ]
}
```

Memungkinkan peran profil instans untuk beralih ke peran dalam akun lain

Anda dapat mengizinkan aplikasi yang berjalan pada EC2 instans Amazon untuk menjalankan perintah di akun lain. Untuk melakukan ini, Anda harus mengizinkan peran EC2 instans Amazon di akun pertama untuk beralih ke peran di akun kedua.

Bayangkan Anda menggunakan dua Akun AWS dan Anda ingin mengizinkan aplikasi yang berjalan pada EC2 instance Amazon untuk berjalan AWS CLIperintah di kedua akun. Asumsikan bahwa EC2 instance Amazon ada di akun`111111111111`. Instance tersebut menyertakan peran profil `abcd` instance yang memungkinkan aplikasi melakukan tugas Amazon S3 hanya-baca di bucket dalam akun `amzn-s3-demo-bucket1` yang sama. `111111111111` Namun demikian, aplikasi tersebut juga harus diizinkan untuk menjalankan `efgh` peran lintas akun untuk mengakses `amzn-s3-demo-bucket2` bucket Amazon S3 dalam akun `222222222222`.



Peran profil EC2 instans `abcd` Amazon harus memiliki kebijakan izin berikut untuk mengizinkan aplikasi mengakses bucket `amzn-s3-demo-bucket1` Amazon S3:

Akun 111111111111 Kebijakan ***abcd*** Izin Peran

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

`abcd`Peran tersebut harus mempercayai EC2 layanan Amazon untuk mengambil peran tersebut. Untuk melakukannya, peran `abcd` harus memiliki kebijakan kepercayaan berikut:

Akun 111111111111 Kebijakan ***abcd*** Kepercayaan Peran

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
}
```

Anggap bahwa `efgh` peran lintas akun memungkinkan tugas Amazon S3 hanya baca di bucket `amzn-s3-demo-bucket2` dengan akun `222222222222` yang sama. Untuk melakukannya, peran lintas akun `efgh` harus memiliki kebijakan izin berikut:

Akun 222222222222 Kebijakan ***efgh*** Izin Peran

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

Peran `efgh` harus mempercayai profil instans `abcd` untuk menjalankannya. Untuk melakukannya, peran `efgh` harus memiliki kebijakan kepercayaan berikut:

Akun 222222222222 Kebijakan ***efgh*** Kepercayaan Peran

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Bagaimana saya memulainya?

Untuk memahami cara kerja peran dengan EC2 instans Amazon, Anda perlu menggunakan IAM konsol untuk membuat peran, meluncurkan EC2 instance Amazon yang menggunakan peran tersebut, lalu memeriksa instance yang sedang berjalan. Anda dapat menguji metadata instans untuk melihat bagaimana kredensial sementara peran tersedia untuk sebuah instans. Anda juga dapat melihat bagaimana aplikasi yang berjalan pada suatu instans dapat menggunakan peran tersebut. Gunakan sumber daya berikut ini untuk mempelajari lebih banyak.

-
- SDKpenelusuran. Bagian AWS SDKdokumentasi menyertakan penelusuran yang menunjukkan aplikasi yang berjalan di EC2 instans Amazon yang menggunakan kredensi sementara untuk peran untuk membaca bucket Amazon S3. Setiap penelusuran berikut menyajikan langkah serupa dengan bahasa pemrograman yang berbeda:
  - Konfigurasikan IAM Peran untuk Amazon EC2 dengan SDK for Java di AWS SDK for Java Panduan Pengembang
  - Luncurkan EC2 Instans Amazon menggunakan SDK for. NETdi AWS SDK for .NET Panduan Pengembang
  - Membuat EC2 Instans Amazon dengan SDK for Ruby di AWS SDK for Ruby Panduan Pengembang

Informasi terkait

Untuk informasi selengkapnya tentang membuat peran atau peran untuk EC2 instans Amazon, lihat informasi berikut:

- Untuk informasi selengkapnya tentang penggunaan IAM peran dengan EC2 instans Amazon, buka Panduan EC2 Pengguna Amazon.
- Untuk membuat peran, lihat IAMpenciptaan peran
- Untuk informasi lebih lanjut tentang membuat kredensial keamanan sementara, lihat Kredensi keamanan sementara di IAM.
- Jika Anda bekerja dengan IAM API atauCLI, Anda harus membuat dan mengelola profil IAM instance. Untuk informasi selengkapnya tentang profil instans, lihat Gunakan profil contoh.
- Untuk informasi selengkapnya tentang kredensil keamanan sementara untuk peran dalam metadata instans, lihat Mengambil Kredensil Keamanan dari Metadata Instance di Panduan Pengguna Amazon. EC2

Gunakan profil contoh

Gunakan profil instance untuk meneruskan IAM peran ke sebuah EC2 instance. Untuk informasi selengkapnya, lihat IAMperan untuk Amazon EC2 di Panduan EC2 Pengguna Amazon.

Mengelola profil instans (konsol)

Jika Anda menggunakan AWS Management Console untuk membuat peran untuk AmazonEC2, konsol secara otomatis membuat profil instance dan memberinya nama yang sama dengan peran tersebut. Saat menggunakan EC2 konsol Amazon untuk meluncurkan instance dengan IAM peran, Anda dapat memilih peran yang akan dikaitkan dengan instance. Dalam konsol, daftar yang ditampilkan sebenarnya adalah daftar nama profil instance. Konsol tidak membuat profil instance untuk peran yang tidak terkait dengan AmazonEC2.

Anda dapat menggunakan AWS Management Console untuk menghapus IAM peran dan profil instance untuk Amazon EC2 jika peran dan profil instans memiliki nama yang sama. Untuk mempelajari selengkapnya tentang menghapus profil instans, lihat Hapus peran atau profil contoh.

Mengelola profil instance (AWS CLI atau AWS API)

Jika Anda mengelola peran Anda dari AWS CLI atau AWS API, Anda membuat peran dan profil instance sebagai tindakan terpisah. Karena peran dan profil instans dapat memiliki nama yang

berbeda, Anda harus mengetahui nama profil instans Anda serta nama peran yang dimuat olehnya. Dengan begitu Anda dapat memilih profil instance yang benar saat meluncurkan EC2 instance.

Anda dapat melampirkan tag ke IAM sumber daya Anda, termasuk profil instans, untuk mengidentifikasi, mengatur, dan mengontrol akses ke sumber daya tersebut. Anda dapat menandai profil instance hanya ketika Anda menggunakan AWS CLI atau AWS API.

> **Note**
>
> Profil instance hanya dapat berisi satu IAM peran, meskipun peran dapat disertakan dalam beberapa profil instance. Batasan satu peran per profil instans ini tidak dapat ditingkatkan. Anda dapat menghapus peran yang ada dan kemudian menambahkan peran yang berbeda ke profil instans. Anda kemudian harus menunggu perubahan muncul di semua AWS karena konsistensi akhirnya. Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Mengelola profil instans (AWS CLI)

Anda dapat menggunakan AWS CLI perintah berikut untuk bekerja dengan profil instance di AWS akun.

- Buat profil instans: `aws iam create-instance-profile`
- Tandai profil instans: `aws iam tag-instance-profile`
- Cantumkan tanda untuk profil instans: `aws iam list-instance-profile-tags`
- Hapus tanda profil instans: `aws iam untag-instance-profile`
- Tambahkan peran ke profil instans: `aws iam add-role-to-instance-profile`
- Buat daftar profil instans: `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Dapatkan informasi tentang profil instans: `aws iam get-instance-profile`
- Hapus peran dari profil instans: `aws iam remove-role-from-instance-profile`
- Hapus profil instans: `aws iam delete-instance-profile`

Anda juga dapat melampirkan peran ke EC2 instance yang sudah berjalan dengan menggunakan perintah berikut. Untuk informasi selengkapnya, lihat IAMPeran untuk Amazon EC2.

- Lampirkan profil instance dengan peran ke EC2 instance yang berhenti atau berjalan: `aws ec2 associate-iam-instance-profile`
- Dapatkan informasi tentang profil instance yang dilampirkan ke EC2 instance: `aws ec2 describe-iam-instance-profile-associations`
- Lepaskan profil instance dengan peran dari EC2 instance yang berhenti atau berjalan: `aws ec2 disassociate-iam-instance-profile`

Mengelola profil instance (AWS API)

Anda dapat memanggil AWS API operasi berikut untuk bekerja dengan profil instance di file Akun AWS.

- Buat profil instans: `CreateInstanceProfile`
- Tandai profil instans: `TagInstanceProfile`
- Cantumkan tanda di profil instans: `ListInstanceProfileTags`
- Hapus tanda profil instans: `UntagInstanceProfile`
- Tambahkan peran ke profil instans: `AddRoleToInstanceProfile`
- Buat daftar profil instans: `ListInstanceProfiles`, `ListInstanceProfilesForRole`
- Dapatkan informasi tentang profil instans: `GetInstanceProfile`
- Hapus peran dari profil instans: `RemoveRoleFromInstanceProfile`
- Hapus profil instans: `DeleteInstanceProfile`

Anda juga dapat melampirkan peran ke EC2 instance yang sudah berjalan dengan memanggil operasi berikut. Untuk informasi selengkapnya, lihat IAMPeran untuk Amazon EC2.

- Lampirkan profil instance dengan peran ke EC2 instance yang berhenti atau berjalan: `AssociateIamInstanceProfile`
- Dapatkan informasi tentang profil instance yang dilampirkan ke EC2 instance: `DescribeIamInstanceProfileAssociations`
- Lepaskan profil instance dengan peran dari EC2 instance yang berhenti atau berjalan: `DisassociateIamInstanceProfile`

Penyedia dan federasi identitas

Sebagai praktik terbaik, kami menyarankan Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS sumber daya alih-alih membuat IAM pengguna individu di Anda Akun AWS. Dengan penyedia identitas (iDP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk menggunakan AWS sumber daya di akun Anda. Ini berguna jika organisasi Anda sudah memiliki sistem identitas sendiri, seperti direktori pengguna korporat. Ini juga berguna jika Anda membuat aplikasi seluler atau aplikasi web yang membutuhkan akses ke AWS sumber daya.

> **Note**
>
> Anda juga dapat mengelola pengguna manusia di Pusat IAM Identitas dengan penyedia SAML identitas eksternal alih-alih menggunakan SAML federasi diIAM. IAMFederasi Pusat Identitas dengan penyedia identitas menyediakan kemampuan bagi Anda untuk memberi orang akses ke beberapa AWS akun di organisasi Anda dan ke beberapa AWS aplikasi. Untuk informasi tentang situasi tertentu di mana IAM pengguna diperlukan, lihat Kapan membuat IAM pengguna (bukan peran).

Jika Anda lebih suka menggunakan satu AWS akun tanpa mengaktifkan Pusat IAM Identitas, Anda dapat menggunakan IAM dengan IDP eksternal yang menyediakan informasi identitas AWS untuk menggunakan OpenID OIDC Connect (SAML) atau 2.0 (Security Assertion Markup Language 2.0). OIDCmenghubungkan aplikasi, seperti GitHub Actions, yang tidak berjalan AWS ke AWS sumber daya. Contoh penyedia SAML identitas terkenal adalah Shibboleth dan Layanan Federasi Direktori Aktif.

Saat Anda menggunakan penyedia identitas , Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. IdP menyediakannya untuk Anda. Pengguna eksternal Anda masuk melalui iDP, dan Anda dapat memberikan izin identitas eksternal tersebut untuk menggunakan AWS sumber daya di akun Anda. Penyedia identitas membantu menjaga Akun AWS keamanan Anda karena Anda tidak perlu mendistribusikan atau menanamkan kredensi keamanan jangka panjang, seperti kunci akses, dalam aplikasi Anda.

Tinjau tabel berikut untuk membantu menentukan jenis IAM federasi mana yang terbaik untuk kasus penggunaan Anda;IAM, Pusat IAM Identitas, atau Amazon Cognito. Ringkasan dan tabel berikut memberikan gambaran umum tentang metode yang dapat digunakan pengguna Anda untuk mendapatkan akses gabungan ke sumber daya. AWS

IAMjenis federasi	Jenis akun	Manajemen akses..	Sumber identitas yang didukung
Federasi dengan Pusat IAM Identitas	Beberapa akun dikelola oleh AWS Organizations	Pengguna manusia tenaga kerja Anda	• SAML 2.0 • Direktori Aktif Terkelola • Direktori Pusat Identitas
Federasi dengan IAM	Akun tunggal dan mandiri	• Pengguna manusia dalam penerapan jangka pendek dan skala kecil • Pengguna mesin	• SAML 2.0 • OIDC
Federasi dengan kumpulan identitas Amazon Cognito	Setiap	Pengguna aplikasi yang memerlukan IAM otorisasi untuk mengakses sumber daya	• SAML 2.0 • OIDC • Pilih OAuth 2.0 penyedia identitas sosial

Federasi dengan Pusat IAM Identitas

Untuk manajemen akses terpusat pengguna manusia, kami sarankan Anda menggunakan Pusat IAM Identitas untuk mengelola akses ke akun dan izin Anda dalam akun tersebut. Pengguna di Pusat IAM Identitas diberikan kredensi jangka pendek ke sumber daya Anda AWS . Anda dapat menggunakan Active Directory, penyedia identitas eksternal (iDP), atau direktori Pusat IAM Identitas sebagai sumber identitas bagi pengguna dan grup untuk menetapkan akses ke sumber daya Anda. AWS

IAMIdentity Center mendukung federasi identitas dengan SAML (Security Assertion Markup Language) 2.0 untuk menyediakan akses masuk tunggal federasi bagi pengguna yang berwenang untuk menggunakan aplikasi dalam portal akses. AWS Pengguna kemudian dapat masuk tunggal ke layanan yang mendukungSAML, termasuk aplikasi pihak ketiga AWS Management Console dan aplikasi, seperti Microsoft 365, SAP Concur, dan Salesforce.

Federasi dengan IAM

Meskipun kami sangat menyarankan untuk mengelola pengguna manusia di Pusat IAM Identitas, Anda dapat mengaktifkan akses pengguna gabungan IAM untuk pengguna manusia dalam penerapan skala kecil jangka pendek. IAMmemungkinkan Anda untuk menggunakan SAML 2.0 terpisah dan Open ID Connect (OIDC) IdPs dan menggunakan atribut pengguna federasi untuk kontrol akses. DenganIAM, Anda dapat meneruskan atribut pengguna, seperti pusat biaya, judul, atau lokal, dari IdPs ke Anda AWS, dan menerapkan izin akses berbutir halus berdasarkan atribut ini.

Beban kerja adalah kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi atau proses backend. Beban kerja Anda dapat memerlukan IAM identitas untuk membuat permintaan ke AWS layanan, aplikasi, alat operasional, dan komponen. Identitas ini mencakup mesin yang berjalan di AWS lingkungan Anda, seperti EC2 instans atau AWS Lambda fungsi Amazon.

Anda juga dapat mengelola identitas mesin untuk pihak eksternal yang membutuhkan akses. Untuk memberikan akses ke identitas mesin, Anda dapat menggunakan IAM peran. IAMperan memiliki izin khusus dan menyediakan cara untuk mengakses AWS dengan mengandalkan kredenal keamanan sementara dengan sesi peran. Selain itu, Anda mungkin memiliki mesin di luar AWS yang membutuhkan akses ke AWS lingkungan Anda. Untuk mesin yang berjalan di luar AWS Anda dapat menggunakan IAMPeran Di Mana Saja. Untuk informasi lebih lanjut tentang peran, lihat IAMperan. Untuk detail tentang cara menggunakan peran untuk mendelegasikan akses di seluruh Akun AWS, lihatIAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran.

Untuk menautkan IDP secara langsung keIAM, Anda membuat entitas penyedia identitas untuk membangun hubungan kepercayaan antara Anda Akun AWS dan iDP. IAMmendukung IdPs yang kompatibel dengan OpenID Connect (OIDC) atau SAML 2.0 (Security Assertion Markup Language 2.0). Untuk informasi selengkapnya tentang menggunakan salah satu dari ini IdPs dengan AWS, lihat bagian berikut:

- OIDCfederasi
- SAML2.0 federasi

Federasi dengan kumpulan identitas Amazon Cognito

Amazon Cognito dirancang untuk pengembang yang ingin mengautentikasi dan mengotorisasi pengguna di aplikasi seluler dan web mereka. Kumpulan pengguna Amazon Cognito menambahkan fitur masuk dan pendaftaran ke aplikasi Anda, dan kumpulan identitas memberikan IAM kredensi

yang memberi pengguna akses ke sumber daya terlindungi yang Anda kelola. AWS Identity pool memperoleh kredensi untuk sesi sementara melalui operasi. `AssumeRoleWithWebIdentity`API

Amazon Cognito bekerja dengan penyedia identitas eksternal yang mendukung dan SAML OpenID Connect, dan dengan penyedia identitas sosial seperti Facebook, Google, dan Amazon. Aplikasi Anda dapat masuk ke pengguna dengan kumpulan pengguna atau iDP eksternal, lalu mengambil sumber daya atas nama mereka dengan sesi sementara yang disesuaikan dalam peran. IAM

Sumber daya tambahan

- Untuk demonstrasi tentang cara membuat proxy federasi kustom yang memungkinkan single sign-on (SSO) ke dalam AWS Management Console menggunakan sistem autentikasi organisasi Anda, lihat. Aktifkan akses broker identitas khusus ke AWS konsol

Skenario umum

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat IAM Identitas untuk mengelola akses ke beberapa secara terpusat Akun AWS dan memberi pengguna akses masuk tunggal yang MFA dilindungi ke semua akun yang ditetapkan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di Pusat IAM Identitas atau dengan mudah terhubung ke penyedia identitas kompatibel SAML 2.0 yang ada. Untuk informasi lebih lanjut, lihat Apa itu Pusat IAM Identitas? dalam AWS IAM Identity Center User Guide.

Anda dapat menggunakan penyedia identitas eksternal (iDP) untuk mengelola identitas pengguna di luar AWS. dan iDP eksternal. IdP eksternal dapat memberikan informasi identitas untuk AWS menggunakan OpenID Connect (OIDC) atau Security Assertion Markup Language (). SAML OIDCumumnya digunakan ketika aplikasi yang tidak berjalan AWS membutuhkan akses ke AWS sumber daya.

Ketika Anda ingin mengkonfigurasi federasi dengan iDP eksternal, Anda membuat penyedia IAM identitas untuk menginformasikan AWS tentang iDP eksternal dan konfigurasinya. Ini membangun

kepercayaan antara IDP Anda Akun AWS dan eksternal. Topik berikut menyediakan skenario umum untuk menggunakan penyedia IAM identitas.

Topik

- Amazon Cognito untuk aplikasi seluler
- OIDCfederasi untuk aplikasi seluler

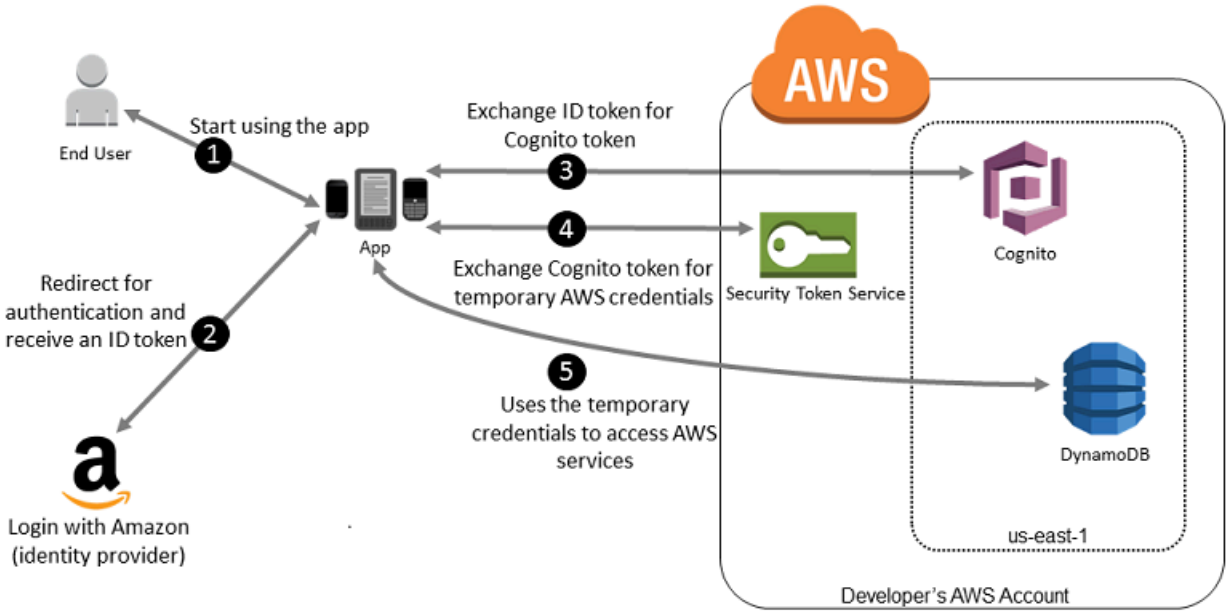
Amazon Cognito untuk aplikasi seluler

Cara yang lebih disukai untuk menggunakan OIDC federasi adalah dengan menggunakan Amazon Cognito. Misalnya, Adele pengembang sedang membangun game untuk perangkat seluler di mana data pengguna seperti skor dan profil disimpan di Amazon S3 dan Amazon DynamoDB. Adele juga dapat menyimpan data ini secara lokal di perangkat dan menggunakan Amazon Cognito untuk tetap disinkronkan di seluruh perangkat. Ia tahu bahwa untuk alasan keamanan dan pemeliharaan, kredensial keamanan jangka panjang AWS tidak boleh didistribusikan bersama dengan game. Dia juga tahu bahwa game akan memiliki banyak sekali pengguna. Untuk semua alasan ini, dia tidak ingin membuat identitas pengguna baru IAM untuk setiap pemain. Sebagai gantinya, dia membangun game sehingga pengguna dapat masuk menggunakan identitas yang telah mereka buat dengan penyedia identitas eksternal (IDP) yang terkenal, seperti Login with Amazon, Facebook, Google, atau iDP yang kompatibel dengan OpenID Connect OIDC (). Game ini bisa memanfaatkan mekanisme autentikasi dari salah satu penyedia ini untuk memvalidasi identitas pengguna.

Untuk mengaktifkan aplikasi seluler mengakses AWS sumber dayanya, Adele pertama-tama mendaftar untuk ID pengembang dengan pilihannya. IdPs Dia juga mengonfigurasi aplikasi dengan setiap penyedia ini. Dalam dirinya Akun AWS yang berisi bucket Amazon S3 dan tabel DynamoDB untuk game, Adele menggunakan Amazon Cognito untuk membuat IAM peran yang secara tepat menentukan izin yang dibutuhkan game. Jika dia menggunakan OIDC iDP, dia juga menciptakan entitas penyedia IAM OIDC identitas untuk membangun kepercayaan antara kumpulan identitas Amazon Cognito dalam dirinya Akun AWS dan iDP.

Dalam kode aplikasi, Adele memanggil antarmuka masuk untuk IdP yang sudah dikonfigurasi sebelumnya. IDP menangani semua detail membiarkan pengguna masuk, dan aplikasi mendapatkan token OAuth akses atau token OIDC ID dari penyedia. Aplikasi Adele dapat menukar informasi otentikasi ini dengan serangkaian kredensi keamanan sementara yang terdiri dari ID kunci AWS akses, kunci akses rahasia, dan token sesi. Aplikasi kemudian dapat menggunakan kredensi ini untuk mengakses layanan web yang ditawarkan oleh. AWS Aplikasi ini terbatas dengan izin yang ditentukan dalam peran yang sudah diasumsikan.

Gambar berikut menunjukkan alur cara kerja yang disederhanakan, menggunakan Login with Amazon sebagai IdP. Untuk Langkah 2, aplikasi ini juga dapat menggunakan Facebook, Google, atau iDP OIDC yang kompatibel, tetapi itu tidak ditampilkan di sini.



1. Seorang pelanggan memulai aplikasi Anda di perangkat seluler. Aplikasi meminta pengguna untuk masuk.
2. Aplikasi menggunakan sumber daya Login with Amazon untuk menerima kredensial pengguna.
3. Aplikasi ini menggunakan API operasi Amazon Cognito `GetId` dan `GetCredentialsForIdentity` untuk menukar token Login with Amazon ID dengan token Amazon Cognito. Amazon Cognito, yang telah dikonfigurasi untuk mempercayai proyek Login with Amazon Anda, menghasilkan token yang ditukar dengan kredensi sesi sementara. AWS STS
4. Aplikasi ini menerima kredensi keamanan sementara dari Amazon Cognito. Aplikasi Anda juga dapat menggunakan alur kerja Dasar (Klasik) di Amazon Cognito untuk mengambil token dari penggunaan. AWS STS `AssumeRoleWithWebIdentity` Untuk informasi selengkapnya, lihat Alur autentikasi kumpulan identitas (identitas gabungan) di Panduan Pengembang Amazon Cognito.
5. Kredensial keamanan sementara dapat digunakan oleh aplikasi untuk mengakses setiap sumber daya AWS yang diperlukan oleh aplikasi untuk beroperasi. Peran yang terkait dengan kredensi keamanan sementara dan kebijakan yang ditetapkan menentukan apa yang dapat diakses.

Gunakan proses berikut untuk mengonfigurasi aplikasi agar menggunakan Amazon Cognito guna mengautentikasi pengguna dan memberikan akses aplikasi ke resource. AWS Untuk langkah-langkah khusus untuk menyelesaikan skenario ini, lihat dokumentasi untuk Amazon Cognito.

1. (Opsional) Daftar sebagai pengembang dengan Login with Amazon, Facebook, Google, atau OpenID Connect (OIDC) lainnya —IdP yang kompatibel dan konfigurasikan satu atau beberapa aplikasi dengan penyedia. Langkah ini bersifat opsional karena Amazon Cognito juga mendukung akses tidak terotentikasi (tamu) untuk pengguna Anda.
2. Pergi ke Amazon Cognito di. AWS Management Console Gunakan wizard Amazon Cognito untuk membuat kumpulan identitas, yang merupakan wadah yang digunakan Amazon Cognito untuk menjaga identitas pengguna akhir tetap tertata untuk aplikasi Anda. Anda dapat berbagi kolam identitas di antara aplikasi. Saat Anda menyiapkan kumpulan identitas, Amazon Cognito membuat satu atau dua IAM peran (satu untuk identitas yang diautentikasi, dan satu untuk identitas “tamu” yang tidak diautentikasi) yang menentukan izin untuk pengguna Amazon Cognito.
3. Integrasikan AWSAmplify dengan aplikasi Anda, dan impor file yang diperlukan untuk menggunakan Amazon Cognito.
4. Buat instance penyedia kredensi Amazon Cognito, meneruskan ID kumpulan identitas, Akun AWS nomor Anda, dan Nama Sumber Daya Amazon (ARN) dari peran yang Anda kaitkan dengan kumpulan identitas. Wizard Amazon Cognito di AWS Management Console menyediakan kode contoh untuk membantu Anda memulai.
5. Saat aplikasi Anda mengakses AWS resource, teruskan instance penyedia kredensial ke objek klien, yang meneruskan kredensi keamanan sementara ke klien. Izin untuk kredensial didasarkan pada peran atau peran-peran yang Anda tetapkan sebelumnya.

Untuk informasi selengkapnya, lihat berikut ini:

- Masuk (Android) di Dokumentasi AWS Amplify Kerangka.
- Masuk (iOS) di Dokumentasi AWS Amplify Kerangka.

OIDCfederasi untuk aplikasi seluler

Untuk hasil terbaik, gunakan Amazon Cognito sebagai broker identitas Anda untuk hampir semua skenario OIDC federasi. Amazon Cognito mudah digunakan dan menyediakan kemampuan tambahan seperti akses anonim (tidak terotentikasi), dan sinkronisasi data pengguna di seluruh perangkat dan penyedia. Namun, jika Anda telah membuat aplikasi yang menggunakan OIDC

federasi dengan memanggil secara manual `AssumeRoleWithWebIdentity`API, Anda dapat terus menggunakannya dan aplikasi Anda akan tetap berfungsi dengan baik.

Proses untuk menggunakan OIDC federasi tanpa Amazon Cognito mengikuti garis besar umum ini:

1. Daftarkan diri Anda sebagai developer dengan penyedia identitas eksternal (IdP) dan konfigurasi aplikasi Anda dengan IdP, yang memberikan ID unik untuk aplikasi Anda. (Berbeda IdPs menggunakan terminologi yang berbeda untuk proses ini. Garis besar ini menggunakan istilah konfigurasi untuk proses identifikasi aplikasi Anda dengan iDP.) Setiap iDP memberi Anda ID aplikasi yang unik untuk IDP tersebut, jadi jika Anda mengonfigurasi aplikasi yang sama dengan beberapa aplikasi IdPs, aplikasi Anda akan memiliki beberapa aplikasi. IDs Anda dapat mengonfigurasi beberapa aplikasi dengan setiap penyedia.

   Tautan eksternal berikut memberikan informasi tentang penggunaan beberapa penyedia identitas yang umum digunakan (IdPs):

   - Login dengan Amazon Developer Center
   - Tambahkan Login Facebook ke Aplikasi atau Website Anda di situs pengembang Facebook.
   - Menggunakan OAuth 2.0 untuk Login (OpenID Connect) di situs pengembang Google.

   > **Important**
   >
   > Jika Anda menggunakan penyedia OIDC identitas dari Google, Facebook, atau Amazon Cognito, jangan membuat penyedia IAM identitas terpisah di. AWS Management Console AWS memiliki penyedia OIDC identitas ini bawaan dan tersedia untuk Anda gunakan. Lewati langkah ini dan langsung buat peran baru menggunakan penyedia identitas Anda.

2. Jika Anda menggunakan iDP selain Google, Facebook, atau Amazon Cognito yang kompatibel OIDC dengannya, buat IAM entitas penyedia identitas untuknya.
3. DiIAM, buat satu atau lebih peran. Untuk setiap peran, tentukan siapa yang dapat mengambil peran (kebijakan kepercayaan) dan izin apa yang dimiliki pengguna aplikasi (kebijakan izin). Biasanya, Anda membuat satu peran untuk setiap IdP yang mendukung aplikasi. Misalnya, Anda dapat membuat peran yang diambil oleh aplikasi jika pengguna masuk melalui Login with Amazon, peran kedua untuk aplikasi yang sama jika pengguna masuk melalui Facebook, dan peran ketiga untuk aplikasi jika pengguna masuk melalui Google. Untuk hubungan kepercayaan, tentukan IdP (seperti Amazon.com) sebagai `Principal` (entitas tepercaya), dan sertakan `Condition` yang sesuai dengan ID aplikasi yang ditetapkan IdP. Contoh peran untuk penyedia yang berbeda dijelaskan dalam Buat peran untuk penyedia identitas pihak ketiga (federasi).

4. Dalam aplikasi, autentikasi pengguna Anda dengan IdP. Spesifikasi tentang cara melakukan ini bervariasi baik menurut IDP yang Anda gunakan (Login with Amazon, Facebook, atau Google) dan di platform mana aplikasi Anda berjalan. Misalnya, metode otentikasi aplikasi Android dapat berbeda dari metode aplikasi iOS atau aplikasi web JavaScript berbasis.

   Biasanya, jika pengguna belum masuk, IdP menampilkan halaman masuk. Setelah IdP mengautentikasi pengguna, IdP mengembalikan informasi dengan token autentikasi tentang pengguna ke aplikasi Anda. Informasi yang disertakan tergantung pada apa yang diekspos IdP dan informasi apa yang ingin dibagikan oleh pengguna. Anda dapat menggunakan informasi ini di aplikasi Anda.

5. Di aplikasi Anda, buatlah panggilan berhenti menandatangani ke `AssumeRoleWithWebIdentity` tindakan untuk meminta kredensial keamanan sementara. Dalam permintaan, Anda meneruskan token autentikasi IDP dan menentukan Amazon Resource Name (ARN) untuk IAM peran yang Anda buat untuk IDP tersebut. AWS memverifikasi bahwa token tersebut tepercaya dan valid dan jika demikian, mengembalikan kredensi keamanan sementara ke aplikasi Anda yang memiliki izin untuk peran yang Anda beri nama dalam permintaan. Jawaban juga mencakup metadata tentang pengguna dari IdP, seperti ID pengguna unik yang dikaitkan IdP dengan pengguna.

6. Menggunakan kredenal keamanan sementara dari `AssumeRoleWithWebIdentity` respons, aplikasi Anda membuat permintaan yang ditandatangani ke AWS API operasi. Informasi ID pengguna dari iDP dapat membedakan pengguna di aplikasi Anda. Misalnya, Anda dapat memasukkan objek ke folder Amazon S3 yang menyertakan ID pengguna sebagai awalan atau sufiks. Hal ini memungkinkan Anda membuat kebijakan kontrol akses yang mengunci folder sehingga hanya pengguna dengan ID tersebut yang tidak dapat mengaksesnya. Untuk informasi selengkapnya, lihat AWS STS prinsip sesi pengguna federasi.

7. Aplikasi Anda harus menyimpan kredensial keamanan sementara sehingga Anda tidak perlu mendapatkan kredensial baru setiap kali aplikasi perlu membuat permintaan ke AWS. Secara default, kredensial akan berlaku selama satu jam. Saat kredensial kedaluwarsa (atau sebelum itu), Anda melakukan panggilan lain ke `AssumeRoleWithWebIdentity` untuk mendapatkan satu set kredensial keamanan sementara baru. Tergantung pada IdP dan cara mereka mengelola token, Anda mungkin harus menyegarkan token IdP sebelum melakukan panggilan baru ke`AssumeRoleWithWebIdentity` , karena token IdP juga biasanya kedaluwarsa setelah waktu tetap. Jika Anda menggunakan AWS SDK untuk iOS atau Android, Anda dapat menggunakan tindakan mazonSTSCredentialsPenyedia A, yang mengelola kredensi IAM sementara, termasuk menyegarkannya sesuai kebutuhan. AWS SDK

OIDCfederasi

Bayangkan Anda membuat aplikasi yang mengakses AWS sumber daya, seperti GitHub Tindakan yang menggunakan alur kerja untuk mengakses Amazon S3 dan DynamoDB.

Saat Anda menggunakan alur kerja ini, Anda membuat permintaan ke AWS layanan yang harus ditandatangani dengan kunci AWS akses. Namun, kami sangat menyarankan agar Anda tidak menyimpan AWS kredensi jangka panjang dalam aplikasi di luar. AWSSebagai gantinya, konfigurasikan aplikasi Anda untuk meminta kredensi AWS keamanan sementara secara dinamis saat diperlukan menggunakan federasi. OIDC Kredensi sementara yang disediakan memetakan ke AWS peran yang hanya memiliki izin yang diperlukan untuk melakukan tugas yang diperlukan oleh aplikasi.

Dengan OIDC federasi, Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Sebagai gantinya, Anda dapat menggunakan OIDC dalam aplikasi, seperti GitHub Actions atau IdP lain yang kompatibel dengan OpenID Connect (OIDC), untuk mengautentikasi dengan. AWS Mereka menerima token otentikasi, yang dikenal sebagai Token JSON Web (JWT), dan kemudian menukar token itu dengan kredensi keamanan sementara di peta AWS itu ke IAM peran dengan izin untuk menggunakan sumber daya tertentu di peta Anda. Akun AWS Menggunakan IDP membantu Anda menjaga Akun AWS keamanan karena Anda tidak perlu menanamkan dan mendistribusikan kredensi keamanan jangka panjang dengan aplikasi Anda.

Dalam sebagian besar skenario, kami menyarankan agar Anda menggunakan Amazon Cognito karena sebagai pialang identitas dan melakukan banyak pekerjaan federasi untuk Anda. Untuk detailnya, lihat bagian berikut, Amazon Cognito untuk aplikasi seluler.

> **Note**
>
> JSONWeb Tokens (JWTs) yang dikeluarkan oleh penyedia identitas OpenID Connect (OIDC) berisi waktu kedaluwarsa dalam `exp` klaim yang menentukan kapan token kedaluwarsa. IAMmenyediakan jendela lima menit di luar waktu kedaluwarsa yang ditentukan dalam akun JWT to untuk kemiringan jam, seperti yang diizinkan oleh standar OpenID Connect () Core 1.0. OIDC Ini berarti OIDC JWTs diterima IAM setelah waktu kedaluwarsa tetapi dalam jendela lima menit ini diterima untuk evaluasi dan pemrosesan lebih lanjut.

Topik

- Sumber daya tambahan untuk OIDC federasi

- Buat penyedia identitas OpenID Connect (OIDC) di IAM
- Dapatkan cap jempol untuk penyedia identitas OpenID Connect

Sumber daya tambahan untuk OIDC federasi

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang OIDC federasi:

- Menggunakan OpenID Connect dalam GitHub alur kerja Anda dengan Mengonfigurasi OpenID Connect di Amazon Web Services
- Identitas Amazon Cognito di Amplify Libraries for Android Guide dan Amazon Cognito Identity di Amplify Libraries for Swift Guide.
- Mengotomatisasi Peran Identitas AWS IAM Web Berbasis Koneksi OpenID dengan Microsoft Entra ID di AWS blog Jaringan Mitra APN () berjalan melalui cara mengautentikasi proses latar belakang otomatis atau aplikasi yang berjalan di luar penggunaan otorisasi. AWS machine-to-machine OIDC
- Artikel Federasi Identitas Web dengan Aplikasi Seluler membahas OIDC federasi dan menunjukkan contoh bagaimana menggunakan OIDC federasi untuk mendapatkan akses ke konten di Amazon S3.

Buat penyedia identitas OpenID Connect (OIDC) di IAM

IAMOIDCpenyedia identitas adalah entitas IAM yang mendeskripsikan layanan penyedia identitas eksternal (iDP) yang mendukung standar OpenID Connect (OIDC), seperti Google atau Salesforce. Anda menggunakan penyedia IAM OIDC identitas ketika Anda ingin membangun kepercayaan antara iDP OIDC -kompatibel dan Anda. Akun AWS Ini berguna saat membuat aplikasi seluler atau aplikasi web yang memerlukan akses ke AWS sumber daya, tetapi Anda tidak ingin membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Untuk informasi selengkapnya tentang skenario ini, lihat the section called “OIDCfederasi”.

Anda dapat membuat dan mengelola penyedia IAM OIDC identitas menggunakan AWS Management Console, AWS Command Line Interface, Alat untuk Windows PowerShell, atau IAMAPI.

Setelah Anda membuat penyedia IAM OIDC identitas, Anda harus membuat satu atau beberapa IAM peran. Peran adalah identitas AWS yang tidak memiliki kredensialnya sendiri (seperti yang dilakukan pengguna). Namun dalam konteks ini, peran ditetapkan secara dinamis ke pengguna gabungan yang diautentikasi oleh IdP organisasi Anda. Peran mengizinkan IdP organisasi Anda meminta kredensial keamanan sementara untuk akses ke AWS. Kebijakan yang ditetapkan untuk peran menentukan

apa yang diizinkan dilakukan oleh pengguna federasi. AWS Untuk membuat peran bagi penyedia identitas pihak ketiga, lihat Buat peran untuk penyedia identitas pihak ketiga (federasi).

> **Important**
>
> Saat Anda mengonfigurasi kebijakan berbasis identitas untuk tindakan yang mendukung `oidc-provider` sumber daya, IAM evaluasi penyedia OIDC identitas lengkapURL, termasuk jalur yang ditentukan. Jika penyedia OIDC identitas Anda URL memiliki jalur, Anda harus menyertakan jalur itu dalam nilai `oidc-provider` ARN sebagai `Resource` elemen. Anda juga memiliki opsi untuk menambahkan garis miring ke depan dan wildcard (`/*`) ke URL domain atau menggunakan karakter wildcard (*dan?) di titik mana pun di jalur. URL Jika penyedia OIDC identitas URL dalam permintaan tidak cocok dengan nilai yang ditetapkan dalam `Resource` elemen kebijakan, permintaan gagal.

Untuk memecahkan masalah umum dengan IAM OIDC federasi, lihat Menyelesaikan kesalahan yang terkait dengan OIDC pada AWS re:Post.

Topik

- Prasyarat: Validasi konfigurasi penyedia identitas Anda
- Membuat dan mengelola OIDC penyedia (konsol)
- Membuat dan mengelola penyedia IAM OIDC identitas (AWS CLI)
- Membuat dan mengelola Penyedia OIDC Identitas (AWS API)

Prasyarat: Validasi konfigurasi penyedia identitas Anda

Sebelum Anda dapat membuat penyedia IAM OIDC identitas, Anda harus memiliki informasi berikut dari IDP Anda. Untuk informasi selengkapnya tentang mendapatkan Informasi konfigurasi OIDC penyedia, lihat dokumentasi untuk IDP Anda.

1. Tentukan penyedia OIDC identitas Anda yang tersedia URL untuk umum. URLHarus dimulai dengan https://. Per the OIDC standard, path com ponent diperbolehkan tetapi parameter kueri tidak. Biasanya, hanya URL terdiri dari nama host, seperti https://server.example.org or https://example.com. URLSeharusnya tidak berisi nomor port.
2. Tambahkan /.well-known/openid-configuration ke akhir penyedia OIDC identitas Anda untuk melihat dokumen konfigurasi dan metadata penyedia yang tersedia URL untuk umum. Anda

harus memiliki dokumen penemuan dalam JSON format dengan dokumen konfigurasi penyedia dan metadata yang dapat diambil dari titik akhir penemuan penyedia OpenID Connect. URL

3. Konfirmasikan bahwa nilai berikut disertakan dalam informasi konfigurasi penyedia Anda. Jika konfigurasi terbuka Anda tidak memiliki salah satu bidang ini, Anda harus memperbarui dokumen penemuan Anda. Proses ini dapat bervariasi berdasarkan penyedia identitas Anda, jadi ikuti dokumentasi IDP Anda untuk menyelesaikan tugas ini.

   - penerbit: URL Untuk domain Anda.
   - jwks_uri: Titik akhir JSON Web Key Set (JWKS) tempat IAM mendapatkan kunci publik Anda. Penyedia identitas Anda harus menyertakan titik akhir JSON Web Key Set (JWKS) dalam konfigurasi openid. Ini URI menentukan di mana mendapatkan kunci publik Anda yang digunakan untuk memverifikasi token yang ditandatangani dari penyedia identitas Anda.
   - claims_supported: Informasi tentang pengguna yang membantu Anda memastikan respons OIDC autentikasi dari IDP berisi atribut wajib yang AWS digunakan dalam IAM kebijakan untuk memeriksa izin bagi pengguna federasi. Untuk daftar kunci IAM kondisi yang dapat digunakan untuk klaim, lihatKunci yang tersedia untuk AWS OIDC federasi.
     - aud: Anda harus menentukan nilai klaim audiens masalah IDP Anda di JSON Web Tokens ()JWTs. Klaim audiens (aud) adalah aplikasi khusus dan mengidentifikasi penerima token yang dituju. Saat Anda mendaftarkan aplikasi seluler atau web dengan penyedia OpenID Connect, mereka membuat ID klien yang mengidentifikasi aplikasi. ID klien adalah pengenal unik untuk aplikasi Anda yang diteruskan dalam klaim aud untuk autentikasi. Klaim aud harus sesuai dengan nilai Audiens saat membuat penyedia IAM OIDC identitas Anda.
     - iat: Klaim harus menyertakan nilai untuk `iat` yang mewakili waktu token ID dikeluarkan.
     - iss: URL Penyedia identitas. URLHarus dimulai dengan https:// and should correspond to the Provider URL provided to IAM. Per the OIDC standard, path com ponent diperbolehkan tetapi parameter kueri tidak. Biasanya, hanya URL terdiri dari nama host, seperti https://server.example.org or https://example.com. URLSeharusnya tidak berisi nomor port.
   - response_types_supported: id_token
   - subject_types_supported: publik
   - id_token_signing_alg_values_supported: RS256

   > **Note**
   >
   > Anda dapat menyertakan klaim tambahan seperti kustom dalam contoh di bawah ini; namun, AWS STS akan mengabaikan klaim tersebut.

```
{
  "issuer": "https://example-domain.com",
  "jwks_uri": "https://example-domain.com/jwks/keys",
  "claims_supported": [
    "aud",
    "iat",
    "iss",
    "name",
    "sub",
    "custom"
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256"
  ],
  "subject_types_supported": [
    "public"
  ]
}
```

Membuat dan mengelola OIDC penyedia (konsol)

Ikuti petunjuk ini untuk membuat dan mengelola penyedia IAM OIDC identitas di AWS Management Console.

> **Important**
>
> Jika Anda menggunakan penyedia OIDC identitas dari Google, Facebook, atau Amazon Cognito, jangan membuat penyedia IAM identitas terpisah menggunakan prosedur ini. Penyedia OIDC identitas ini sudah terpasang AWS dan tersedia untuk Anda gunakan. Alih-alih, ikuti langkah-langkah untuk membuat peran baru bagi penyedia identitas Anda, lihat Buat peran untuk federasi OpenID Connect (konsol).

Untuk membuat penyedia IAM OIDC identitas (konsol)

1. Sebelum Anda membuat penyedia IAM OIDC identitas, Anda harus mendaftarkan aplikasi Anda dengan IDP untuk menerima ID klien. ID klien (juga dikenal sebagai audiens) adalah pengenal unik untuk aplikasi yang diberikan kepada Anda saat mendaftarkan aplikasi dengan IdP. Untuk informasi lebih lanjut tentang mendapatkan ID klien, lihat dokumentasi untuk IdP.

   > **Note**
   >
   > AWS mengamankan komunikasi dengan penyedia OIDC identitas (IdPs) menggunakan pustaka otoritas sertifikat root tepercaya (CAs) kami untuk memverifikasi sertifikat titik akhir JSON Web Key Set (JWKS). TLS Jika OIDC IDP Anda bergantung pada sertifikat yang tidak ditandatangani oleh salah satu dari tepercaya iniCAs, maka kami mengamankan komunikasi menggunakan cap jempol yang diatur dalam konfigurasi iDP. AWS akan kembali ke verifikasi sidik jari jika kami tidak dapat mengambil TLS sertifikat atau jika TLS v1.3 diperlukan.

2. Buka IAM konsol di https://console.aws.amazon.com/iam/.
3. Di panel navigasi, pilih Penyedia identitas, lalu pilih Tambahkan penyedia.
4. Untuk Konfigurasi penyedia, pilih OpenID Connect.
5. Untuk Provider URL, URL ketik IDP. URLHarus mematuhi batasan ini:

   - URLIni peka huruf besar/kecil.
   - URLHarus dimulai dengan**`https://`**.
   - URLSeharusnya tidak berisi nomor port.
   - Di dalam Anda Akun AWS, setiap penyedia IAM OIDC identitas harus menggunakan yang unikURL. Jika Anda mencoba mengirimkan URL yang telah digunakan untuk penyedia OpenID Connect di Akun AWS, Anda akan mendapatkan kesalahan.

6. Untuk Audiens, ketikkan ID klien dari aplikasi yang Anda daftarkan dengan IDP dan terimaStep 1, dan yang membuat permintaan. AWS Jika Anda memiliki klien tambahan IDs (juga dikenal sebagai audiens) untuk IDP ini, Anda dapat menambahkannya nanti di halaman detail penyedia.

   > **Note**
   >
   > Jika JWT token IDP Anda menyertakan `azp` klaim, masukkan nilai ini sebagai nilai Audiens.

Jika penyedia OIDC identitas Anda menetapkan keduanya `aud` dan `azp` klaim dalam token, AWS STS akan menggunakan nilai dalam `azp` klaim sebagai `aud` klaim.

7. (Opsional) Untuk Menambahkan tag, Anda dapat menambahkan pasangan kunci-nilai untuk membantu Anda mengidentifikasi dan mengatur. IdPs Anda juga dapat menggunakan tanda untuk mengontrol akses ke sumber daya AWS . Untuk mempelajari lebih lanjut tentang menandai penyedia IAM OIDC identitas, lihatTandai OpenID Connect (OIDC) penyedia identitas. Pilih Tambahkan tanda. Masukkan nilai untuk setiap pasangan nilai-kunci tanda.
8. Verifikasi informasi yang telah Anda berikan. Setelah selesai, pilih Tambahkan penyedia. IAMakan mencoba mengambil dan menggunakan cap jempol CA perantara teratas dari sertifikat server iDP OIDC untuk membuat penyedia identitas. IAM OIDC

> **Note**
>
> Rantai sertifikat penyedia OIDC identitas harus dimulai dengan domain atau penerbitURL, kemudian sertifikat perantara, dan diakhiri dengan sertifikat root. Jika urutan rantai sertifikat berbeda atau menyertakan duplikat atau sertifikat tambahan, maka Anda menerima kesalahan ketidakcocokan tanda tangan dan STS gagal memvalidasi Token JSON Web (). JWT Perbaiki urutan sertifikat dalam rantai yang dikembalikan dari server untuk menyelesaikan kesalahan. Untuk informasi selengkapnya tentang standar rantai sertifikat, lihat certificate_list di RFC 5246 di situs web Seri. RFC

9. Tetapkan IAM peran ke penyedia identitas Anda untuk memberikan identitas pengguna eksternal yang dikelola oleh izin penyedia identitas Anda untuk mengakses AWS sumber daya di akun Anda. Untuk mempelajari lebih lanjut tentang menciptakan peran untuk federasi identitas, lihat Buat peran untuk penyedia identitas pihak ketiga (federasi).

> **Note**
>
> OIDC IdPs digunakan dalam kebijakan kepercayaan peran harus dalam akun yang sama dengan peran yang mempercayainya.

Untuk menambah atau menghapus sidik jari untuk penyedia IAM OIDC identitas (konsol)

> **Note**
>
> AWS mengamankan komunikasi dengan penyedia OIDC identitas (IdPs) menggunakan pustaka otoritas sertifikat root tepercaya (CAs) kami untuk memverifikasi sertifikat titik akhir JSON Web Key Set (JWKS). TLS Jika OIDC IDP Anda bergantung pada sertifikat yang tidak ditandatangani oleh salah satu dari tepercaya iniCAs, maka kami mengamankan komunikasi menggunakan cap jempol yang diatur dalam konfigurasi iDP. AWS akan kembali ke verifikasi sidik jari jika kami tidak dapat mengambil TLS sertifikat atau jika TLS v1.3 diperlukan.

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, silakan pilih Penyedia identitas. Kemudian pilih nama penyedia IAM identitas yang ingin Anda perbarui.
3. Pilih tab Verifikasi titik akhir, lalu di bagian Cetak Jempol, pilih Kelola. Untuk memasukkan nilai sidik jari baru, pilih Tambahkan sidik jari. Untuk menghapus sidik jari, pilih Hapus di samping sidik jari yang ingin Anda hapus.

   > **Note**
   >
   > Penyedia IAM OIDC identitas harus memiliki setidaknya satu dan dapat memiliki maksimal lima sidik jari.

   Setelah selesai, pilih Simpan perubahan.

Untuk menambahkan audiens untuk penyedia IAM OIDC identitas (konsol)

1. Di panel navigasi, pilih Penyedia identitas, lalu pilih nama penyedia IAM identitas yang ingin Anda perbarui.
2. Di bagian Audiens, pilih Tindakan dan pilih Tambahkan audiens.
3. Ketik ID klien dari aplikasi yang Anda daftarkan dengan IDP dan diterimaStep 1, dan itu akan membuat permintaan untuk. AWS Lalu, pilih Tambahkan audiens.

> **Note**
>
> Penyedia IAM OIDC identitas harus memiliki setidaknya satu dan dapat memiliki maksimal 100 pemirsa.

Untuk menghapus audiens untuk penyedia IAM OIDC identitas (konsol)

1. Di panel navigasi, pilih Penyedia identitas, lalu pilih nama penyedia IAM identitas yang ingin Anda perbarui.
2. Di bagian Audiens, pilih tombol radio di samping audiens yang ingin Anda hapus, lalu pilih Tindakan.
3. Pilih Hapus audiens. Jendela baru terbuka.
4. Jika Anda menghapus audiens, identitas yang bergabung dengan audiens tidak dapat mengambil peran yang terkait dengan audiens. Di jendela, baca peringatan dan konfirmasikan bahwa Anda ingin menghapus audiens dengan mengetikkan kata `remove` di bidangnya.
5. Pilih Hapus untuk menghapus audiens.

Untuk menghapus penyedia IAM OIDC identitas (konsol)

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, silakan pilih Penyedia identitas.
3. Pilih kotak centang di samping penyedia IAM identitas yang ingin Anda hapus. Jendela baru terbuka.
4. Konfirmasikan bahwa Anda ingin menghapus penyedia dengan mengetik kata `delete` di bidangnya. Lalu, pilih Hapus.

Membuat dan mengelola penyedia IAM OIDC identitas (AWS CLI)

Anda dapat menggunakan AWS CLI perintah berikut untuk membuat dan mengelola penyedia IAM OIDC identitas.

Untuk membuat penyedia IAM OIDC identitas (AWS CLI)

1. (Opsional) Untuk mendapatkan daftar semua penyedia IAM OIDC identitas di AWS akun Anda, jalankan perintah berikut:

    - `aws iam list-open-id-connect-providers`

2. Untuk membuat penyedia IAM OIDC identitas baru, jalankan perintah berikut:

    - `aws iam create-open-id-connect-provider`

Untuk memperbarui daftar cap jempol sertifikat server untuk penyedia IAM OIDC identitas yang ada ()AWS CLI

- Untuk memperbarui daftar cap jempol sertifikat server untuk penyedia IAM OIDC identitas, jalankan perintah berikut:

    - `aws iam update-open-id-connect-provider-thumbprint`

Untuk menandai penyedia IAM OIDC identitas yang ada (AWS CLI)

- Untuk menandai penyedia IAM OIDC identitas yang ada, jalankan perintah berikut:

    - `aws iam tag-open-id-connect-provider`

Untuk mencantumkan tag untuk penyedia IAM OIDC identitas yang ada (AWS CLI)

- Untuk membuat daftar tag untuk penyedia IAM OIDC identitas yang ada, jalankan perintah berikut:

    - `aws iam list-open-id-connect-provider-tags`

Untuk menghapus tag pada penyedia IAM OIDC identitas (AWS CLI)

- Untuk menghapus tag pada penyedia IAM OIDC identitas yang ada, jalankan perintah berikut:

    - `aws iam untag-open-id-connect-provider`

Untuk menambah atau menghapus ID klien dari penyedia IAM OIDC identitas yang ada (AWS CLI)

1. (Opsional) Untuk mendapatkan daftar semua penyedia IAM OIDC identitas di AWS akun Anda, jalankan perintah berikut:

    - `aws iam list-open-id-connect-providers`

2. (Opsional) Untuk mendapatkan informasi rinci tentang penyedia IAM OIDC identitas, jalankan perintah berikut:

   - `aws iam get-open-id-connect-provider`
3. Untuk menambahkan ID klien baru ke penyedia IAM OIDC identitas yang ada, jalankan perintah berikut:

   - `aws iam add-client-id-to-open-id-connect-provider`
4. Untuk menghapus klien dari penyedia IAM OIDC identitas yang ada, jalankan perintah berikut:

   - `aws iam remove-client-id-from-open-id-connect-provider`

Untuk menghapus penyedia IAM OIDC identitas (AWS CLI)

1. (Opsional) Untuk mendapatkan daftar semua penyedia IAM OIDC identitas di AWS akun Anda, jalankan perintah berikut:

   - `aws iam list-open-id-connect-providers`
2. (Opsional) Untuk mendapatkan informasi rinci tentang penyedia IAM OIDC identitas, jalankan perintah berikut:

   - `aws iam get-open-id-connect-provider`
3. Untuk menghapus penyedia IAM OIDC identitas, jalankan perintah berikut:

   - `aws iam delete-open-id-connect-provider`

Membuat dan mengelola Penyedia OIDC Identitas (AWS API)

Anda dapat menggunakan IAM API perintah berikut untuk membuat dan mengelola OIDC penyedia.

Untuk membuat penyedia IAM OIDC identitas (AWS API)

1. (Opsional) Untuk mendapatkan daftar semua penyedia IAM OIDC identitas di AWS akun Anda, hubungi operasi berikut:

   - `ListOpenIDConnectProviders`
2. Untuk membuat penyedia IAM OIDC identitas baru, hubungi operasi berikut:

   - `CreateOpenIDConnectProvider`

Untuk memperbarui daftar cap jempol sertifikat server untuk penyedia IAM OIDC identitas yang ada ()AWS API

- Untuk memperbarui daftar cap jempol sertifikat server untuk penyedia IAM OIDC identitas, hubungi operasi berikut:
  - `UpdateOpenIDConnectProviderThumbprint`

Untuk menandai penyedia IAM OIDC identitas yang ada (AWS API)

- Untuk menandai penyedia IAM OIDC identitas yang ada, hubungi operasi berikut:
  - `TagOpenIDConnectProvider`

Untuk mencantumkan tag untuk penyedia IAM OIDC identitas yang ada (AWS API)

- Untuk mencantumkan tag untuk penyedia IAM OIDC identitas yang ada, panggil operasi berikut:
  - `ListOpenIDConnectProviderTags`

Untuk menghapus tag pada penyedia IAM OIDC identitas yang ada (AWS API)

- Untuk menghapus tag pada penyedia IAM OIDC identitas yang ada, panggil operasi berikut:
  - `UntagOpenIDConnectProvider`

Untuk menambah atau menghapus ID klien dari penyedia IAM OIDC identitas yang ada (AWS API)

1. (Opsional) Untuk mendapatkan daftar semua penyedia IAM OIDC identitas di AWS akun Anda, hubungi operasi berikut:
   - `ListOpenIDConnectProviders`
2. (Opsional) Untuk mendapatkan informasi rinci tentang penyedia IAM OIDC identitas, hubungi operasi berikut:
   - `GetOpenIDConnectProvider`
3. Untuk menambahkan ID klien baru ke penyedia IAM OIDC identitas yang ada, panggil operasi berikut ini:

- `AddClientIDToOpenIDConnectProvider`

4. Untuk menghapus ID klien dari penyedia IAM OIDC identitas yang ada, panggil operasi berikut:

   - `RemoveClientIDFromOpenIDConnectProvider`

Untuk menghapus penyedia IAM OIDC identitas (AWS API)

1. (Opsional) Untuk mendapatkan daftar semua penyedia IAM OIDC identitas di AWS akun Anda, hubungi operasi berikut:

   - `ListOpenIDConnectProviders`

2. (Opsional) Untuk mendapatkan informasi rinci tentang penyedia IAM OIDC identitas, hubungi operasi berikut:

   - `GetOpenIDConnectProvider`

3. Untuk menghapus penyedia IAM OIDC identitas, hubungi operasi berikut:

   - `DeleteOpenIDConnectProvider`

Dapatkan cap jempol untuk penyedia identitas OpenID Connect

Saat Anda membuat penyedia identitas OpenID Connect (OIDC) di IAM, IAM memerlukan cap jempol untuk otoritas sertifikat perantara (CA) teratas yang menandatangani sertifikat yang digunakan oleh penyedia identitas eksternal (iDP). Sidik jari adalah tanda tangan untuk sertifikat CA yang digunakan untuk menerbitkan sertifikat bagi IdP yang kompatibel dengan OIDC. Saat Anda membuat penyedia identitas IAM OIDC, Anda mempercayai identitas yang diautentikasi oleh IDP tersebut untuk memiliki akses ke identitas Anda. Akun AWS Dengan menggunakan cap jempol sertifikat CA, Anda mempercayai sertifikat apa pun yang dikeluarkan oleh CA tersebut dengan nama DNS yang sama dengan yang terdaftar. Ini menghilangkan perlunya memperbarui kepercayaan di setiap akun ketika Anda memperpanjang sertifikat tanda tangan IdP.

> **Important**
>
> Dalam kebanyakan kasus, server federasi menggunakan dua sertifikat berbeda:

- Yang pertama membuat koneksi HTTPS antara AWS dan IDP Anda. Ini harus dikeluarkan oleh CA root publik yang terkenal, seperti AWS Certificate Manager. Ini memungkinkan klien untuk memeriksa keandalan dan status sertifikat.
- Yang kedua digunakan untuk mengenkripsi token, dan harus ditandatangani oleh CA root pribadi atau publik.

Anda dapat membuat penyedia identitas IAM OIDC dengan, Tools for Windows AWS Command Line Interface PowerShell, atau IAM API. Saat Anda menggunakan metode ini, Anda memiliki opsi untuk memberikan sidik jari secara manual. Jika Anda memilih untuk tidak menyertakan sidik jari, IAM akan mengambil cap jempol CA menengah atas dari sertifikat server OIDC iDP. Jika Anda memilih untuk menyertakan sidik jari, Anda harus mendapatkan sidik jari secara manual dan menyediakannya. AWS

Saat Anda membuat penyedia identitas OIDC dengan konsol IAM, IAM mencoba mengambil cap jempol CA perantara teratas dari sertifikat server OIDC iDP untuk Anda.

Kami menyarankan Anda juga mendapatkan sidik jari untuk OIDC iDP Anda secara manual dan memverifikasi bahwa IAM mengambil sidik jari yang benar. Untuk informasi selengkapnya tentang mendapatkan cap jempol sertifikat, lihat bagian berikut.

> **Note**
>
> AWS mengamankan komunikasi dengan penyedia identitas OIDC (IdPs) menggunakan perpustakaan otoritas sertifikat root tepercaya (CA) kami untuk memverifikasi sertifikat TLS titik akhir JSON Web Key Set (JWKS). Jika IDP OIDC Anda bergantung pada sertifikat yang tidak ditandatangani oleh salah satu CA tepercaya ini, maka kami mengamankan komunikasi menggunakan cap jempol yang diatur dalam konfigurasi IDP. AWS akan kembali ke verifikasi sidik jari jika kami tidak dapat mengambil sertifikat TLS atau jika TLS v1.3 diperlukan.

Dapatkan cap jempol sertifikat

Anda menggunakan browser web dan alat baris perintah OpenSSL untuk mendapatkan cap jempol sertifikat untuk penyedia OIDC. Namun, Anda tidak perlu mendapatkan cap jempol sertifikat secara manual untuk membuat penyedia identitas IAM OIDC. Anda dapat menggunakan prosedur berikut untuk mendapatkan cap jempol sertifikat dari penyedia OIDC Anda.

Untuk mendapatkan sidik jari untuk IdP OIDC

1. Sebelum mendapatkan sidik jari untuk IdP OIDC, Anda harus mendapatkan alat baris perintah OpenSSL. Anda menggunakan alat ini untuk mengunduh rantai sertifikat IdP OIDC dan menghasilkan sidik jari untuk sertifikat akhir dalam rantai sertifikat. Jika Anda perlu menginstal dan mengonfigurasi OpenSSL, ikuti petunjuk di Instal OpenSSL dan Konfigurasi OpenSSL.
2. Mulailah dengan URL IdP OIDC (misalnya, `https://server.example.com`), lalu tambahkan `/.well-known/openid-configuration` untuk membentuk URL bagi dokumen konfigurasi IdP, seperti berikut:

   `https://server.example.com/.well-known/openid-configuration`

   Buka URL ini di browser web, ganti *`server.example.com`* dengan nama server IdP Anda.
3. Pada dokumen yang ditampilkan, gunakan fitur Cari pada browser web untuk menemukan teks `"jwks_uri"`. Setelah teks `"jwks_uri"`, terdapat tanda titik dua (:) diikuti oleh URL. Salin nama domain yang sepenuhnya memenuhi syarat dari URL. Jangan sertakan `https://` atau jalur apa pun yang mengikuti domain tingkat atas.

   ```
   {
    "issuer": "https://accounts.example.com",
    "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
    "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
    "token_endpoint": "https://oauth2.exampleapis.com/token",
    "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
    "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
    "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
   ...
   ```
4. Gunakan alat baris perintah OpenSSL untuk menjalankan perintah berikut. Ganti *`keys.example.com`* dengan nama domain yang Anda peroleh di Step 3.

   ```
   openssl s_client -servername keys.example.com -showcerts -
   connect keys.example.com:443
   ```
5. Di jendela perintah Anda, gulir ke atas hingga Anda melihat sertifikat yang serupa dengan contoh berikut. Jika Anda melihat lebih dari satu sertifikat, cari sertifikat terakhir yang ditampilkan (di bagian akhir output perintah). Ini berisi sertifikat CA menengah teratas dalam rantai otoritas sertifikat.

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
 nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
 NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
 -----END CERTIFICATE-----
```

Menyalin sertifikat (termasuk baris `-----BEGIN CERTIFICATE-----` dan `-----END CERTIFICATE-----`) dan tempel ke dalam file teks. Lalu simpan file dengan nama file **`certificate.crt`**.

> **Note**
>
> Rantai sertifikat penyedia identitas OIDC harus dimulai dengan domain atau URL penerbit, kemudian sertifikat perantara, dan diakhiri dengan sertifikat root. Jika urutan rantai sertifikat berbeda atau menyertakan duplikat atau sertifikat tambahan, maka Anda menerima kesalahan ketidakcocokan tanda tangan dan STS gagal memvalidasi JSON Web Token (JWT). Perbaiki urutan sertifikat dalam rantai yang dikembalikan dari server untuk menyelesaikan kesalahan. Untuk informasi selengkapnya tentang standar rantai sertifikat, lihat certificate_list di RFC 5246 di situs web Seri RFC.

6. Gunakan alat baris perintah OpenSSL untuk menjalankan perintah berikut.

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

Jendela perintah Anda menampilkan sidik jari sertifikat, yang terlihat mirip dengan contoh berikut:

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Hapus karakter titik dua (:) dari string ini untuk membuat sidik jari akhir, seperti ini:

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

7. Jika Anda membuat penyedia identitas IAM OIDC dengan AWS CLI, Tools for Windows, atau IAM API PowerShell, memberikan cap jempol adalah opsional. Jika Anda memilih untuk tidak menyertakan sidik jari selama pembuatan, IAM akan mengambil cap jempol CA menengah atas dari sertifikat server OIDC iDP. Setelah penyedia identitas IAM OIDC dibuat, Anda dapat membandingkan sidik jari ini dengan sidik jari yang diambil oleh IAM.

   Jika Anda membuat penyedia identitas IAM OIDC di konsol IAM, konsol mencoba mengambil cap jempol CA perantara teratas dari sertifikat server OIDC iDP untuk Anda. Anda dapat membandingkan sidik jari ini dengan sidik jari yang diambil oleh IAM. Setelah penyedia identitas IAM OIDC dibuat, Anda dapat melihat sidik jari untuk penyedia identitas IAM OIDC di tab Verifikasi titik akhir pada halaman konsol Ringkasan penyedia OIDC.

   > **Important**
   >
   > Jika sidik jari yang Anda peroleh tidak cocok dengan yang Anda lihat di detail cap jempol penyedia identitas IAM OIDC, Anda sebaiknya tidak menggunakan penyedia OIDC. Sebagai gantinya, Anda harus menghapus penyedia OIDC yang dibuat dan kemudian mencoba lagi untuk membuat penyedia OIDC setelah beberapa waktu berlalu. Verifikasi bahwa cap jempol cocok sebelum Anda menggunakan penyedia. Jika sidik jari masih belum sama setelah percobaan kedua, gunakan Forum IAM untuk menghubungi AWS.

Instal OpenSSL

Jika Anda belum menginstal OpenSSL, ikuti petunjuk di bagian ini.

Untuk menginstal OpenSSL di Linux atau Unix

1. Pergi ke OpenSSL: Sumber, Tarballs (https://openssl.org/source/).
2. Unduh sumber terbaru dan buat paketnya.

Untuk menginstal OpenSSL di Windows

1. Buka OpenSSL: Distribusi Biner (https://wiki.openssl.org/index.php/Binaries) untuk daftar situs tempat Anda dapat menginstal versi Windows.
2. Ikuti petunjuk di situs yang Anda pilih untuk memulai pemasangan.
3. Jika Anda diminta memasang Microsoft Visual C++ 2008 Redistributables dan itu belum diinstal di sistem Anda, pilih tautan unduhan yang sesuai dengan lingkungan Anda. Ikuti petunjuk yang diberikan oleh Microsoft Visual C++ 2008 Redistributable Setup Wizard.

   > **Note**
   >
   > Jika Anda tidak yakin apakah Microsoft Visual C ++ 2008 Redistributables sudah diinstal pada sistem Anda, Anda dapat mencoba menginstal OpenSSL terlebih dahulu. Installer OpenSSL menampilkan peringatan jika Microsoft Visual C ++ 2008 Redistributables belum diinstal. Pastikan Anda menginstal arsitektur (32-bit atau 64-bit) yang cocok dengan versi OpenSSL yang Anda instal.

4. Setelah Anda menginstal Microsoft Visual C ++ 2008 Redistributables, pilih versi binari OpenSSL yang sesuai untuk lingkungan Anda dan simpan file secara lokal. Mulai OpenSSL Setup Wizard.
5. Ikuti petunjuk yang dijelaskan dalam OpenSSL Setup Wizard.

Konfigurasi OpenSSL

Sebelum Anda menggunakan perintah OpenSSL, Anda harus mengkonfigurasi sistem operasi sehingga memiliki informasi tentang lokasi di mana OpenSSL diinstal.

Untuk mengkonfigurasi OpenSSL di Linux atau Unix

1. Pada baris perintah, atur `OpenSSL_HOME` variabel ke lokasi instalasi OpenSSL:

   ```
   $ export OpenSSL_HOME=path_to_your_OpenSSL_installation
   ```

2. Atur jalur untuk menyertakan instalasi OpenSSL:

   ```
   $ export PATH=$PATH:$OpenSSL_HOME/bin
   ```

> **Note**
>
> Perubahan apa pun yang Anda lakukan terhadap variabel dengan perintah `export` hanya berlaku untuk sesi saat ini. Anda dapat membuat perubahan terus-menerus pada variabel lingkungan dengan menyetelnya di file konfigurasi shell Anda. Untuk informasi lebih lanjut, lihat dokumentasi untuk sistem operasi Anda.

Untuk mengkonfigurasi OpenSSL pada Windows

1. Buka jendela Prompt Perintah.
2. Mengatur `OpenSSL_HOME` variabel ke lokasi instalasi OpenSSL:

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. Atur `OpenSSL_CONF` variabel ke lokasi file konfigurasi di instalasi OpenSSL Anda:

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Atur jalur untuk menyertakan instalasi OpenSSL:

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> Perubahan apa pun yang Anda lakukan terhadap variabel lingkungan Windows dalam jendela Prompt Perintah hanya valid untuk sesi baris perintah saat ini. Anda dapat membuat perubahan persisten pada variabel lingkungan dengan mengaturnya sebagai properti sistem. Prosedur yang tepat tergantung pada versi Windows yang Anda gunakan. (Misalnya, di Windows 7, buka Panel Kontrol, Sistem dan Keamanan, Sistem. Kemudian pilih Pengaturan sistem lanjutan, tab Lanjutan, Variabel Lingkungan.) Untuk informasi selengkapnya, lihat dokumentasi Windows.

SAML2.0 federasi

AWS mendukung federasi identitas dengan SAML2.0 (Security Assertion Markup Language 2.0), standar terbuka yang digunakan oleh banyak penyedia identitas (IdPs). Fitur ini memungkinkan sistem masuk tunggal gabungan (SSO), sehingga pengguna dapat masuk ke AWS Management Console atau menelepon AWS APIoperasi tanpa Anda harus membuat IAM pengguna untuk semua orang di organisasi Anda. Dengan menggunakanSAML, Anda dapat menyederhanakan proses konfigurasi federasi dengan AWS, karena Anda dapat menggunakan layanan IDP alih-alih menulis kode proxy identitas kustom.

Federasi IAM mendukung kasus penggunaan ini:

- Akses federasi untuk memungkinkan pengguna atau aplikasi di organisasi Anda menelepon AWS APIoperasi . Kasus penggunaan ini dibahas di bagian berikut. Anda menggunakan SAML pernyataan (sebagai bagian dari respons autentikasi) yang dihasilkan di organisasi Anda untuk mendapatkan kredensil keamanan sementara. Skenario ini mirip dengan skenario federasi lain yang IAM mendukung, seperti yang dijelaskan dalam Minta kredensi keamanan sementara danOIDCfederasi. Namun, SAML berbasis 2.0 IdPs di organisasi Anda menangani banyak detail saat dijalankan untuk melakukan otentikasi dan pemeriksaan otorisasi.
- Single sign-on () SSO berbasis web ke AWS Management Console dari organisasi Anda . Pengguna dapat masuk ke portal di organisasi Anda yang dihosting oleh iDP yang SAML kompatibel dengan 2.0, pilih opsi untuk masuk AWS, dan diarahkan ke konsol tanpa harus memberikan informasi masuk tambahan. Anda dapat menggunakan SAML iDP pihak ketiga untuk membuat SSO akses ke konsol atau Anda dapat membuat iDP khusus untuk mengaktifkan akses konsol bagi pengguna eksternal Anda. Untuk informasi selengkapnya tentang membangun IdP kustom, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

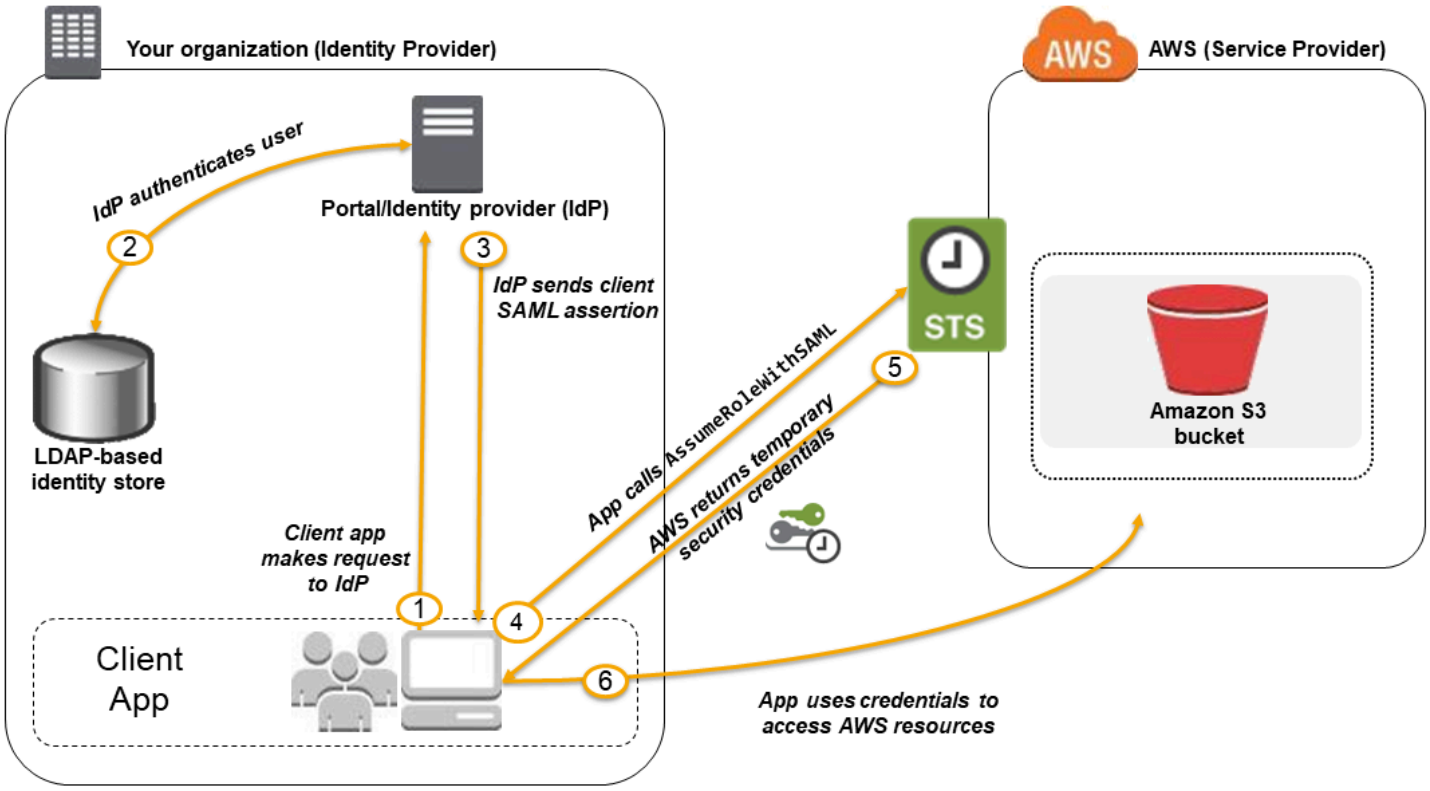
Topik

- Menggunakan federasi SAML berbasis untuk API akses ke AWS
- Ikhtisar konfigurasi federasi berbasis SAML 2.0
- Ikhtisar peran untuk memungkinkan akses SAML -federasi ke Anda AWS sumber daya
- Mengidentifikasi pengguna secara unik di SAML federasi berbasis
- Buat penyedia SAML identitas di IAM
- Konfigurasikan IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim

- Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS
- Konfigurasikan SAML pernyataan untuk respons otentikasi
- Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console
- Lihat SAML respons di browser Anda

Menggunakan federasi SAML berbasis untuk API akses ke AWS

Asumsikan bahwa Anda ingin menyediakan cara bagi karyawan untuk menyalin data dari komputer mereka ke folder cadangan. Anda membuat aplikasi yang dapat dijalankan pengguna di komputer mereka. Di bagian belakang, aplikasi membaca dan menulis objek dalam ember Amazon S3. Pengguna tidak memiliki akses langsung ke AWS. Sebagai gantinya, proses berikut digunakan:



1. Pengguna dalam organisasi Anda menggunakan aplikasi klien untuk meminta autentikasi dari IdP organisasi Anda.
2. IdP mengautentikasi pengguna menggunakan penyimpanan identitas organisasi Anda.
3. IdP membuat SAML pernyataan dengan informasi tentang pengguna dan mengirimkan pernyataan ke aplikasi klien.

4. Aplikasi klien memanggil AWS STS `AssumeRoleWithSAML`API, melewati SAML penyedia, peran yang akan ARN diasumsikan, dan SAML pernyataan dari IDP. ARN
5. APIRespons terhadap aplikasi klien mencakup kredensil keamanan sementara.
6. Aplikasi klien menggunakan kredensil keamanan sementara untuk memanggil operasi Amazon API S3.

Ikhtisar konfigurasi federasi berbasis SAML 2.0

Sebelum Anda dapat menggunakan federasi SAML berbasis 2.0 seperti yang dijelaskan dalam skenario dan diagram sebelumnya, Anda harus mengonfigurasi IDP organisasi Anda dan Akun AWS untuk saling percaya. Proses umum untuk mengonfigurasikan kepercayaan ini dijelaskan dalam langkah-langkah berikut. Di dalam organisasi Anda, Anda harus memiliki IDP yang mendukung SAML 2.0, seperti Microsoft Active Directory Federation Service (AD FS, bagian dari Windows Server), Shibboleth, atau penyedia 2.0 lain yang kompatibel. SAML

> **Note**
>
> Untuk meningkatkan ketahanan federasi, kami sarankan Anda mengonfigurasi IDP dan AWS federasi untuk mendukung beberapa SAML titik akhir masuk. Untuk detailnya, lihat AWS Artikel Blog Keamanan Cara menggunakan SAML endpoint regional untuk failover.

Konfigurasikan iDP organisasi Anda dan AWS untuk saling percaya

1. Pendaftaran AWS sebagai penyedia layanan (SP) dengan IDP organisasi Anda. Gunakan dokumen SAML metadata dari `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

   Untuk daftar kemungkinan *`region-code`* nilai, lihat kolom Region di AWS Titik akhir masuk.

   Anda dapat secara opsional menggunakan dokumen SAML metadata dari. `https://signin.aws.amazon.com/static/saml-metadata.xml`
2. Menggunakan iDP organisasi Anda, Anda menghasilkan XML file metadata setara yang dapat menggambarkan IDP Anda sebagai penyedia identitas di IAM AWS. Itu harus menyertakan nama penerbit, tanggal pembuatan, tanggal kedaluwarsa, dan kunci yang AWS dapat digunakan untuk memvalidasi tanggapan otentikasi (pernyataan) dari organisasi Anda.

3. Di IAM konsol, Anda membuat penyedia SAML identitas. Sebagai bagian dari proses ini, Anda mengunggah dokumen SAML metadata oleh iDP di organisasi Anda. Step 2 Untuk informasi selengkapnya, lihat Buat penyedia SAML identitas di IAM.
4. DiIAM, Anda membuat satu atau lebih IAM peran. Dalam kebijakan kepercayaan peran, Anda menetapkan SAML penyedia sebagai prinsipal, yang membangun hubungan kepercayaan antara organisasi Anda dan AWS Kebijakan izin peran menetapkan hal yang diizinkan oleh pengguna organisasi Anda di . AWS Untuk informasi selengkapnya, lihat Buat peran untuk penyedia identitas pihak ketiga (federasi).

   > **Note**
   >
   > SAMLIDPsyang digunakan dalam kebijakan kepercayaan peran harus dalam akun yang sama dengan peran tersebut.

5. Di iDP organisasi Anda, Anda menentukan pernyataan yang memetakan pengguna atau grup di organisasi Anda ke peran. IAM Perhatikan bahwa pengguna dan grup yang berbeda di organisasi Anda mungkin memetakan ke IAM peran yang berbeda. Langkah yang tepat untuk melakukan pemetaan bergantung pada IdP apa yang Anda gunakan. Di skenario sebelumnya dari folder Amazon S3 untuk pengguna, mungkin semua pengguna akan memetakan peran yang sama yang memberikan izin Amazon S3. Untuk informasi selengkapnya, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi.

   Jika IDP Anda memungkinkan untuk SSO AWS konsol, maka Anda dapat mengonfigurasi durasi maksimum sesi konsol. Untuk informasi selengkapnya, lihat Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console.
6. Dalam aplikasi yang Anda buat, Anda memanggil AWS Security Token Service `AssumeRoleWithSAML`API, meneruskannya ke SAML penyedia yang Anda buatStep 3, peran untuk mengasumsikan bahwa Anda membuatStep 4, dan SAML pernyataan tentang pengguna saat ini yang Anda dapatkan dari idP Anda. ARN ARN AWS memastikan bahwa permintaan untuk mengambil peran berasal dari idP yang direferensikan di penyedia. SAML

   Untuk informasi lebih lanjut, lihat AssumeRoleWithSAMLdi AWS Security Token Service APIReferensi.
7. Jika permintaan berhasil, akan API mengembalikan satu set kredensi keamanan sementara, yang dapat digunakan aplikasi Anda untuk membuat permintaan yang ditandatangani AWS. Aplikasi Anda memiliki informasi tentang pengguna saat ini dan dapat mengakses folder khusus pengguna di Amazon S3, seperti yang dijelaskan dalam skenario sebelumnya.

Ikhtisar peran untuk memungkinkan akses SAML -federasi ke Anda AWS sumber daya

Peran atau peran yang Anda buat dalam IAM menentukan apa yang diizinkan dilakukan oleh pengguna federasi dari organisasi Anda AWS. Saat Anda membuat kebijakan kepercayaan untuk peran tersebut, Anda menentukan SAML penyedia yang Anda buat sebelumnya sebagai`Principal`. Anda juga dapat membuat cakupan kebijakan kepercayaan dengan a `Condition` untuk mengizinkan hanya pengguna yang cocok dengan SAML atribut tertentu untuk mengakses peran tersebut. Misalnya, Anda dapat menentukan bahwa hanya pengguna yang SAML afiliasinya `staff` (seperti yang ditegaskan oleh https://openidp.feide.no) yang diizinkan untuk mengakses peran, seperti yang diilustrasikan oleh kebijakan contoh berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

> **Note**
>
> SAMLIDPsyang digunakan dalam kebijakan kepercayaan peran harus dalam akun yang sama dengan peran tersebut.

Untuk informasi selengkapnya tentang SAML kunci yang dapat Anda periksa dalam kebijakan, lihatKunci yang tersedia untuk AWS STS federasi SAML berbasis.

Anda dapat menyertakan titik akhir regional untuk `saml:aud` atribut dihttps://*region-code*`.signin.aws.amazon.com/static/saml-metadata.xml`. Untuk daftar kemungkinan *region-code* nilai, lihat kolom Region di AWS Titik akhir masuk.

Untuk kebijakan izin dalam peran tersebut, Anda menentukan izin seperti yang Anda inginkan untuk peran apa pun. Misalnya, jika pengguna dari organisasi Anda diizinkan untuk mengelola instans Amazon Elastic Compute Cloud, Anda harus secara eksplisit mengizinkan tindakan EC2 Amazon dalam kebijakan izin, seperti yang ada dalam kebijakan terkelola Amazon. EC2FullAccess

Mengidentifikasi pengguna secara unik di SAML federasi berbasis

Saat Anda membuat kebijakan akses diIAM, seringkali berguna untuk dapat menentukan izin berdasarkan identitas pengguna. Misalnya, untuk pengguna yang telah menggunakan federasiSAML, aplikasi mungkin ingin menyimpan informasi di Amazon S3 menggunakan struktur seperti ini:

```
amzn-s3-demo-bucket/app1/user1
amzn-s3-demo-bucket/app1/user2
amzn-s3-demo-bucket/app1/user3
```

Anda dapat membuat bucket (`amzn-s3-demo-bucket`) dan folder (`app1`) melalui konsol Amazon S3 atau AWS CLI, karena itu adalah nilai statis. Namun, folder khusus pengguna (*`user1`*, *`user2`*, *`user3`*, dll.) harus dibuat pada waktu pengoperasian menggunakan kode, karena nilai yang mengidentifikasi pengguna tidak diketahui sampai pertama kali pengguna masuk melalui proses federasi.

Untuk menulis kebijakan yang mereferensikan detail khusus pengguna sebagai bagian dari nama sumber daya, identitas pengguna harus tersedia dalam SAML kunci yang dapat digunakan dalam kondisi kebijakan. Kunci berikut tersedia untuk federasi SAML berbasis 2.0 untuk digunakan dalam kebijakan. IAM Anda dapat menggunakan nilai yang dikembalikan oleh kunci berikut untuk membuat pengidentifikasi pengguna unik untuk sumber daya seperti folder Amazon S3.

- `saml:namequalifier`. Nilai hash berdasarkan rangkaian nilai `Issuer` respons (`saml:iss`) dan string dengan ID `AWS` akun dan nama ramah (bagian terakhir dariARN) penyedia di. SAML IAM Gabungan ID akun dan nama ramah SAML penyedia tersedia untuk IAM kebijakan sebagai kuncinya. `saml:doc` ID akun dan nama penyedia harus dipisahkan dengan '/' seperti pada “123456789012/provider_name”. Untuk informasi lebih lanjut, lihat kunci `saml:doc` di Kunci yang tersedia untuk AWS STS federasi SAML berbasis.

  Kombinasi dari `NameQualifier` dan `Subject` dapat digunakan untuk secara unik mengidentifikasi pengguna federasi. Kode pseudo berikut menunjukkan bagaimana nilai ini dihitung. Dalam pseudocode ini + menunjukkan penggabungan, `SHA1` merupakan fungsi yang menghasilkan intisari pesan menggunakan SHA -1, dan `Base64` mewakili fungsi yang menghasilkan versi keluaran hash yang dikodekan Base-64.

```
Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
MySAMLIdP" ) )
```

Untuk informasi selengkapnya tentang kunci kebijakan yang tersedia untuk federasi SAML berbasis, lihatKunci yang tersedia untuk AWS STS federasi SAML berbasis.

- `saml:sub` (string). Ini adalah subjek klaim, yang mencakup nilai yang secara unik mengidentifikasi pengguna secara individu dalam organisasi (misalnya, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (string). Kunci ini bisa`persistent`,`transient`, atau penuh `Format` URI dari `NameID` elemen `Subject` dan yang digunakan dalam SAML pernyataan Anda. Nilai dari `persistent` menunjukkan bahwa nilai dalam `saml:sub` sama untuk pengguna di semua sesi. Jika nilainya `transient`, pengguna memiliki nilai `saml:sub` untuk setiap sesi. Untuk informasi tentang elemen `NameID` `Format` atribut, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi.

Contoh berikut menunjukkan kebijakan izin yang menggunakan kunci sebelumnya untuk memberikan izin ke folder khusus pengguna di Amazon S3. Kebijakan ini mengasumsikan bahwa objek Amazon S3 diidentifikasi menggunakan awalan yang menyertakan keduanya dan. `saml:namequalifier` `saml:sub` Perhatikan bahwa elemen `Condition` mencakup pengujian untuk memastikan bahwa `saml:sub_type` diatur ke `persistent`. Jika diatur ke `transient`, nilai `saml:sub` untuk pengguna dapat berbeda untuk setiap sesi, dan kombinasi nilai tidak boleh digunakan untuk mengidentifikasi folder khusus pengguna.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
```

```
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

Untuk informasi selengkapnya tentang pernyataan pemetaan dari IdP ke kunci kebijakan, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi.

Buat penyedia SAML identitas di IAM

Penyedia identitas IAM SAML 2.0 adalah entitas IAM yang menjelaskan layanan penyedia identitas eksternal (iDP) yang mendukung standar SAML2.0 (Security Assertion Markup Language 2.0). Anda menggunakan penyedia IAM identitas saat ingin membangun kepercayaan antara iDP yang SAML kompatibel seperti Shibboleth atau Layanan Federasi Direktori Aktif AWS dan, sehingga pengguna di organisasi Anda dapat mengakses sumber daya. AWS IAMSAMLPenyedia identitas digunakan sebagai prinsip dalam kebijakan kepercayaan. IAM

Untuk informasi selengkapnya tentang skenario ini, lihat SAML2.0 federasi.

Anda dapat membuat dan mengelola penyedia IAM identitas di AWS Management Console atau dengan AWS CLI, Alat untuk Windows PowerShell, atau AWS API panggilan.

Setelah Anda membuat SAML penyedia, Anda harus membuat satu atau beberapa IAM peran. Peran adalah identitas AWS yang tidak memiliki kredensialnya sendiri (seperti yang dilakukan pengguna). Namun dalam konteks ini, peran ditetapkan secara dinamis ke pengguna gabungan yang diautentikasi oleh IdP organisasi Anda. Peran mengizinkan IdP organisasi Anda meminta kredensial keamanan sementara untuk akses ke AWS. Kebijakan yang ditetapkan untuk peran menentukan apa yang diizinkan dilakukan oleh pengguna federasi. AWS Untuk membuat peran SAML federasi, lihatBuat peran untuk penyedia identitas pihak ketiga (federasi).

Terakhir, setelah Anda membuat peran, Anda menyelesaikan SAML kepercayaan dengan mengonfigurasi IDP Anda dengan informasi AWS tentang dan peran yang Anda inginkan untuk digunakan oleh pengguna federasi Anda. Ini disebut sebagai mengonfigurasi kepercayaan pihak pengandal antara IdP dan AWS. Untuk mengkonfigurasi kepercayaan pihak yang bergantung, lihat Konfigurasikan IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim.

Topik

- Prasyarat
- Membuat dan mengelola penyedia IAM SAML identitas (konsol)

- Membuat dan mengelola Penyedia IAM SAML Identitas (AWS CLI)
- Membuat dan mengelola penyedia IAM SAML identitas (AWS API)
- Langkah selanjutnya

Prasyarat

Sebelum Anda dapat membuat penyedia SAML identitas, Anda harus memiliki informasi berikut dari IDP Anda.

- Dapatkan dokumen SAML metadata dari IDP Anda. Dokumen ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons SAML otentikasi (pernyataan) yang diterima dari iDP. Untuk menghasilkan dokumen metadata, gunakan perangkat lunak manajemen identitas yang disediakan oleh iDP eksternal Anda.

> **Important**
>
> File metadata ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons SAML otentikasi (pernyataan) yang diterima dari iDP. File metadata harus dikodekan dalam format UTF -8 tanpa tanda urutan byte (). BOM Untuk menghapusnyaBOM, Anda dapat menyandikan file sebagai UTF -8 menggunakan alat pengeditan teks, seperti Notepad ++.
> Sertifikat x.509 yang disertakan sebagai bagian dari dokumen SAML metadata harus menggunakan ukuran kunci minimal 1024 bit. Selain itu, sertifikat x.509 juga harus bebas dari ekstensi berulang. Anda dapat menggunakan ekstensi, tetapi ekstensi hanya dapat muncul sekali dalam sertifikat. Jika sertifikat x.509 tidak memenuhi salah satu kondisi, pembuatan IDP gagal dan mengembalikan kesalahan “Tidak dapat mengurai metadata”. Seperti yang didefinisikan oleh Profil Interoperabilitas Metadata SAML V2.0 Versi 1.0, IAM tidak mengevaluasi atau mengambil tindakan terkait berakhirnya sertifikat X.509 dokumen metadata.

Untuk petunjuk tentang cara mengonfigurasi banyak yang tersedia IdPs untuk dikerjakan AWS, termasuk cara membuat dokumen SAML metadata yang diperlukan, lihat. Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS

Untuk bantuan terkait SAML federasi, lihat Pemecahan Masalah SAML federasi.

Membuat dan mengelola penyedia IAM SAML identitas (konsol)

Anda dapat menggunakan AWS Management Console untuk membuat, memperbarui, dan menghapus penyedia IAM SAML identitas. Untuk bantuan terkait SAML federasi, lihat Pemecahan Masalah SAML federasi.

Untuk membuat penyedia IAM SAML identitas (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Penyedia identitas lalu pilih Tambahkan penyedia.
3. Untuk penyedia Konfigurasi, pilih SAML.
4. Ketikkan nama untuk penyedia identitas.
5. Untuk dokumen Metadata, pilih Pilih file, tentukan dokumen SAML metadata yang Anda unduh. the section called “Prasyarat”
6. (Opsional) Untuk Tambahkan tag, Anda dapat menambahkan pasangan kunci-nilai untuk membantu Anda mengidentifikasi dan mengatur. IdPs Anda juga dapat menggunakan tanda untuk mengontrol akses ke sumber daya AWS . Untuk mempelajari lebih lanjut tentang menandai penyedia SAML identitas, lihatTag penyedia IAM SAML identitas.

   Pilih Tambahkan tanda. Masukkan nilai untuk setiap pasangan nilai-kunci tanda.
7. Verifikasi informasi yang telah Anda berikan. Setelah selesai, pilih Tambahkan penyedia.
8. Tetapkan IAM peran ke penyedia identitas Anda untuk memberikan identitas pengguna eksternal yang dikelola oleh izin penyedia identitas Anda untuk mengakses AWS sumber daya di akun Anda. Untuk mempelajari lebih lanjut tentang menciptakan peran untuk federasi identitas, lihat Buat peran untuk penyedia identitas pihak ketiga (federasi).

   > **Note**
   >
   > SAMLIDPsyang digunakan dalam kebijakan kepercayaan peran harus dalam akun yang sama dengan peran tersebut.

Untuk menghapus SAML penyedia (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Penyedia identitas.
3. Pilih tombol radio di samping penyedia identitas yang ingin Anda hapus.
4. Pilih Hapus. Jendela baru terbuka.
5. Konfirmasikan bahwa Anda ingin menghapus penyedia dengan mengetik kata `delete` di bidangnya. Lalu, pilih Hapus.

Membuat dan mengelola Penyedia IAM SAML Identitas (AWS CLI)

Anda dapat menggunakan AWS CLI untuk membuat, memperbarui, dan menghapus SAML penyedia. Untuk bantuan terkait SAML federasi, lihat Pemecahan Masalah SAML federasi.

Untuk membuat penyedia IAM identitas dan mengunggah dokumen metadata ()AWS CLI

- Jalankan perintah ini: `aws iam create-saml-provider`

Untuk memperbarui penyedia IAM SAML identitas (AWS CLI)

- Jalankan perintah ini:`aws iam update-saml-provider`

Untuk menandai penyedia IAM identitas yang ada (AWS CLI)

- Jalankan perintah ini:`aws iam tag-saml-provider`

Untuk mencantumkan tag untuk penyedia IAM identitas yang ada (AWS CLI)

- Jalankan perintah ini:`aws iam list-saml-provider-tags`

Untuk menghapus tag pada penyedia IAM identitas yang ada (AWS CLI)

- Jalankan perintah ini:`aws iam untag-saml-provider`

Untuk menghapus penyedia IAM SAML identitas (AWS CLI)

1. (Opsional) Untuk mencantumkan informasi untuk semua penyedia, seperti tanggal pembuatanARN, dan kedaluwarsa, jalankan perintah berikut:
   - `aws iam list-saml-providers`

2. (Opsional) Untuk mendapatkan informasi tentang penyedia tertentu, seperti, tanggal pembuatanARN, tanggal kedaluwarsa, pengaturan enkripsi, dan informasi kunci pribadi, jalankan perintah berikut:

   - `aws iam get-saml-provider`

3. Untuk menghapus penyedia IAM identitas, jalankan perintah berikut:

   - `aws iam delete-saml-provider`

Membuat dan mengelola penyedia IAM SAML identitas (AWS API)

Anda dapat menggunakan AWS API untuk membuat, memperbarui, dan menghapus SAML penyedia. Untuk bantuan terkait SAML federasi, lihat Pemecahan Masalah SAML federasi.

Untuk membuat penyedia IAM identitas dan mengunggah dokumen metadata ()AWS API

- Hubungi operasi ini: `CreateSAMLProvider`

Untuk memperbarui penyedia IAM SAML identitas (AWS API)

- Hubungi operasi ini: `UpdateSAMLProvider`

Untuk menandai penyedia IAM identitas yang ada (AWS API)

- Hubungi operasi ini: `TagSAMLProvider`

Untuk mencantumkan tag untuk penyedia IAM identitas yang ada (AWS API)

- Hubungi operasi ini: `ListSAMLProviderTags`

Untuk menghapus tag pada penyedia IAM identitas yang ada (AWS API)

- Hubungi operasi ini: `UntagSAMLProvider`

Untuk menghapus penyedia IAM identitas (AWS API)

1. (Opsional) Untuk mencantumkan informasi untuk semua IdPs, sepertiARN, tanggal pembuatan, dan kedaluwarsa, hubungi operasi berikut:

- `ListSAMLProviders`

2. (Opsional) Untuk mendapatkan informasi tentang penyedia tertentu, seperti, tanggal pembuatanARN, tanggal kedaluwarsa, pengaturan enkripsi, dan informasi kunci pribadi, hubungi operasi berikut:

   - `GetSAMLProvider`

3. Untuk menghapus IdP, panggil operasi berikut:

   - `DeleteSAMLProvider`

Langkah selanjutnya

Setelah Anda membuat penyedia SAML identitas, siapkan kepercayaan pihak yang mengandalkan dengan IDP Anda. Anda juga dapat menggunakan klaim dari respons autentikasi IDP Anda dalam kebijakan untuk mengontrol akses ke peran.

- Anda harus memberi tahu IDP tentang AWS sebagai penyedia layanan. Ini disebut menambahkan kepercayaan pihak yang mengandalkan antara AWS IDP Anda dan. Proses yang tepat untuk menambahkan kepercayaan pihak yang bergantung tergantung pada IdP yang Anda gunakan. Untuk detailnya, lihat Konfigurasikan IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim.
- Ketika IDP mengirimkan respons yang berisi klaim AWS, banyak klaim yang masuk memetakan ke AWS kunci konteks. Anda dapat menggunakan kunci konteks ini dalam IAM kebijakan menggunakan elemen Kondisi untuk mengontrol akses ke peran. Untuk detailnya, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi

Konfigurasikan IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim

Saat Anda membuat penyedia identitas IAM dan peran untuk akses SAMP, Anda memberi tahu AWS tentang penyedia identitas eksternal (iDP) dan apa yang boleh dilakukan penggunanya. Langkah Anda selanjutnya adalah memberi tahu IDP tentang AWS sebagai penyedia layanan. Ini disebut menambahkan kepercayaan pihak yang bergantung antara IdP dan AWS. Proses yang tepat untuk menambahkan kepercayaan pihak yang bergantung tergantung pada IdP yang Anda gunakan. Untuk detailnya, lihat dokumentasi untuk perangkat lunak manajemen identitas Anda.

Banyak yang IdPs memungkinkan Anda untuk menentukan URL dari mana IDP dapat membaca dokumen XMLyang berisi informasi dan sertifikat pihak yang bergantung. Untuk AWS, gunakan `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` atau`https://signin.aws.amazon.com/static/saml-metadata.xml`. Untuk daftar kemungkinan nilai *kode wilayah, lihat kolom Region* di titik akhir AWS Masuk.

Jika Anda tidak dapat menentukan URL secara langsung, unduh dokumen XML dari URL sebelumnya dan impor ke perangkat lunak IdP Anda.

Anda juga perlu membuat aturan klaim yang sesuai di IDP Anda yang ditentukan AWS sebagai pihak yang mengandalkan. Ketika IDP mengirimkan respons SAMP ke AWS titik akhir, itu termasuk pernyataan SAMP yang berisi satu atau lebih klaim. Klaim adalah informasi tentang pengguna dan grupnya. Aturan klaim memetakan informasi tersebut ke dalam atribut SAML. Ini memungkinkan Anda memastikan bahwa respons otentikasi SAMP dari IDP Anda berisi atribut yang diperlukan yang AWS digunakan dalam kebijakan IAM untuk memeriksa izin bagi pengguna federasi. Untuk informasi selengkapnya, lihat topik berikut.

- Ikhtisar peran untuk memungkinkan akses SAML -federasi ke Anda AWS sumber daya. Topik ini membahas penggunaan kunci khusus SAML dalam kebijakan IAM dan cara menggunakannya untuk membatasi izin bagi pengguna federasi SAML.
- Konfigurasikan SAML pernyataan untuk respons otentikasi. Topik ini membahas cara mengonfigurasi klaim SAML yang mencakup informasi tentang pengguna. Klaim dikemas menjadi pernyataan SAML dan disertakan dalam respons SAML yang dikirimkan ke AWS. Anda harus memastikan bahwa informasi yang dibutuhkan oleh AWS kebijakan termasuk dalam pernyataan SAMP dalam bentuk yang AWS dapat mengenali dan menggunakan.
- Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS. Topik ini menyediakan tautan ke dokumentasi yang disediakan oleh organisasi pihak ketiga tentang cara mengintegrasikan solusi identitas AWS.

> **Note**
>
> Untuk meningkatkan ketahanan federasi, kami menyarankan Anda mengonfigurasi IDP dan AWS federasi Anda untuk mendukung beberapa titik akhir masuk SAMP. Untuk detailnya, lihat artikel Blog AWS Keamanan Cara menggunakan endpoint SAMP regional untuk failover.

Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensil sementara saat mengakses. AWS Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat Identitas IAM untuk mengelola akses ke beberapa secara terpusat Akun AWS dan memberi pengguna akses masuk tunggal yang dilindungi MFA ke semua akun yang ditetapkan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di IAM Identity Center atau dengan mudah terhubung ke penyedia identitas kompatibel SAMP 2.0 yang ada. Untuk informasi lebih lanjut, lihat Apa itu Pusat Identitas IAM? dalam AWS IAM Identity Center User Guide.

Tautan berikut membantu Anda mengonfigurasi solusi penyedia identitas SAMP 2.0 (IDP) pihak ketiga untuk bekerja AWS dengan federasi.

> **Tip**
>
> AWS Support engineer dapat membantu pelanggan yang memiliki rencana dukungan bisnis dan perusahaan dengan beberapa tugas integrasi yang melibatkan perangkat lunak pihak ketiga. Untuk daftar terkini dari platform dan aplikasi yang didukung, lihat Apa perangkat lunak pihak ketiga yang didukung? dalam FAQ Dukungan AWS .

Solusi	Informasi lebih lanjut
Auth0	Integrasikan dengan Amazon Web Services — Halaman di situs web dokumentasi Auth0 ini memiliki tautan ke sumber daya yang menjelaskan cara mengatur sistem masuk tunggal (SSO) dengan AWS Management Console dan menyertakan contoh. JavaScript Anda dapat mengonfig urasikan Auth0 untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Auth0 Mengumumkan Kemitraan dengan Tag AWS Sesi IAM.

Solusi	Informasi lebih lanjut
Microsoft Entra	Tutorial: Integrasi Microsoft Entra SSO dengan AWS Single-Account Access - Tutorial ini di situs web Microsoft menjelaskan cara mengatur Microsoft Entra (sebelumn ya dikenal sebagai Azure AD) sebagai penyedia identitas (iDP) menggunakan federasi SAMP.
Centrify	Konfigurasikan Centrify dan Gunakan SAMP untuk SSO AWS - Halaman ini di situs web Centrify menjelaskan cara mengonfigurasi Centrify untuk menggunakan SAMP untuk SSO. AWS
CyberArk	Konfigurasikan CyberArkuntuk menyediakan Amazon Web Services (AWS) akses ke pengguna yang masuk melalui SAMP single sign-on (SSO) dari User Portal. CyberArk
ForgeRock	Platform ForgeRock Identitas terintegrasi dengan AWS. Anda dapat mengonfigurasi ForgeRock untuk meneruska n tag sesi. Untuk informasi lebih lanjut, lihat Kontrol Akses Berbasis Atribut untuk Amazon Web Services.
Google Workspace	Aplikasi cloud Amazon Web Services — Artikel di situs Bantuan Admin Google Workspace ini menjelaskan cara mengonfigurasi Google Workspace sebagai IDP SAMP 2.0 sebagai penyedia layanan. AWS
IBM	Anda dapat mengonfigurasikan IBM untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat IBM Cloud Identity IDaaS salah satu yang pertama mendukung tag AWS sesi.
JumpCloud	Memberikan Akses melalui Peran IAM untuk Single Sign On (SSO) dengan Amazon AWS - Artikel ini di JumpCloud situs web menjelaskan cara mengatur dan mengaktifkan SSO berdasarkan peran IAM untuk. AWS

Solusi	Informasi lebih lanjut
Matrix42	MyWorkspace Panduan Memulai — Panduan ini menjelaskan cara mengintegrasikan layanan AWS identitas dengan Matrix42 MyWorkspace.
Microsoft Active Directory Federation Services (AD FS)	Catatan Lapangan: Mengintegrasikan Layanan Federasi Direktori Aktif dengan AWS IAM Identity Center - Posting ini di Blog AWS Arsitektur menjelaskan aliran otentikasi antara AD FS dan AWS IAM Identity Center (IAM Identity Center). IAM Identity Center mendukung federasi identitas dengan SAMP 2.0, memungkinkan integrasi dengan solusi AD FS. Pengguna dapat masuk ke portal Pusat Identitas IAM dengan kredensi perusahaan mereka mengurangi overhead admin untuk mempertahankan kredensil terpisah di IAM Identity Center. Anda juga dapat mengonfigurasi AD FS untuk meneruskan tag sesi. Untuk informasi lebih lanjut, lihat Gunakan kontrol akses berbasis atribut dengan AD FS untuk menyederhanakan manajemen izin IAM.
miniOrange	SSO untuk AWS — Halaman ini di situs web MiniOrange menjelaskan cara membangun akses aman AWS untuk perusahaan dan kontrol penuh atas akses AWS aplikasi.
Okta	Mengintegrasikan Antarmuka Baris Perintah Amazon Web Services Menggunakan Okta - Dari halaman ini di situs dukungan Okta, Anda dapat mempelajari cara mengonfigurasi Okta untuk digunakan. AWS Anda dapat mengonfigurasikan Okta untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Okta dan AWS Mitra untuk Menyederhanakan Akses Melalui Tag Sesi.
Okta	AWS Federasi Akun — Bagian ini di situs web Okta menjelaskan cara mengatur dan mengaktifkan Pusat Identitas IAM untuk. AWS

Solusi	Informasi lebih lanjut
OneLogin	Dari OneLoginKnowledgebase, **SAML AWS** cari daftar artikel yang menjelaskan cara mengatur fungsionalitas Pusat Identitas IAM antara OneLogin dan AWS untuk skenario peran tunggal dan multi-peran. Anda dapat mengonfigurasi OneLogin untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat OneLogin dan Tag Sesi: Kontrol Akses Berbasis Atribut untuk AWS Sumber Daya.
Identitas Ping	PingFederate AWS Konektor - Lihat detail tentang PingFederate AWS Konektor, templat koneksi cepat untuk mengatur koneksi masuk tunggal (SSO) dan penyediaan dengan mudah. Baca dokumentasi dan unduh PingFederate AWS Konektor terbaru untuk integrasi dengan AWS. Anda dapat mengonfigurasikan Identitas Ping untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Mengumumkan Dukungan Identitas Ping untuk Kontrol Akses Berbasis Atribut di AWS.
RadiantLogic	Radiant Logic Technology Partners - Layanan Identitas RadiantOne Federasi Radiant Logic terintegrasi dengan AWS untuk menyediakan pusat identitas untuk SSO berbasis SAMP.
RSA	Amazon Web Services - Panduan Implementasi Siap RSA memberikan panduan untuk mengintegrasikan AWS dan RSA. Untuk informasi selengkapnya tentang konfigurasi SAMP, lihat Amazon Web Services - Konfigurasi SSO Halaman Saya SALL - Panduan Implementasi Siap RSA.
Salesforce.com	Cara mengkonfigurasi SSO dari Salesforce ke AWS - Artikel cara ini di situs pengembang Salesforce.com menjelaskan cara mengatur penyedia identitas (idP) di Salesforce dan mengkonfigurasi sebagai penyedia layanan. AWS

Solusi	Informasi lebih lanjut
SecureAuth	AWS - SecureAuth SAMP SSO - Artikel ini di SecureAuth situs web menjelaskan cara mengatur integrasi SAMP dengan AWS untuk alat. SecureAuth
Shibboleth	Cara Menggunakan Shibboleth untuk SSO ke AWS Management Console - Entri ini di Blog AWS Keamanan menyediakan step-by-step tutorial tentang cara mengatur Shibboleth dan mengkonfigurasinya sebagai penyedia identitas untuk. AWS Anda dapat mengonfigurasikan Shibboleth untuk meneruskan tag sesi.

Untuk detail selengkapnya, lihat halaman Mitra IAM di AWS situs web.

Konfigurasikan SAML pernyataan untuk respons otentikasi

Setelah Anda memverifikasi identitas pengguna di organisasi Anda, penyedia identitas eksternal (iDP) mengirimkan respons otentikasi ke titik akhir di AWS SAML. `https://region-code.signin.aws.amazon.com/saml` Untuk daftar potensi *`region-code`* pengganti, lihat kolom Region di titik akhir AWS Masuk. Respons ini adalah POST permintaan yang menyertakan SAML token yang mematuhi standar HTTPPOSTBinding for SAML 2.0 dan yang berisi elemen, atau klaim berikut. Anda mengonfigurasi klaim ini di iDP SAML yang kompatibel. Lihat dokumentasi IdP Anda untuk instruksi tentang cara memasukkan klaim ini.

Ketika IDP mengirimkan respons yang berisi klaim AWS, banyak klaim yang masuk memetakan ke AWS kunci konteks. Kunci konteks ini dapat diperiksa dalam IAM kebijakan menggunakan `Condition` elemen. Daftar pemetaan yang tersedia mengikuti di bagian ini Memetakan SAML atribut untuk AWS mempercayai kunci konteks kebijakan.

`Subject` dan **`NameID`**

Kebijakan berikut menunjukkan sebuah contoh. Gantikan nilai Anda sendiri untuk nilai yang ditandai. Harus ada tepat satu elemen `SubjectConfirmation` dengan elemen `SubjectConfirmationData` yang menyertakan atribut `NotOnOrAfter`, dan atribut `Recipient`. Atribut ini mencakup nilai yang harus cocok dengan AWS titik akhir`https://region-code.signin.aws.amazon.com/saml`. Untuk daftar kemungkinan *`region-code`* nilai, lihat kolom Region di titik akhir AWS Masuk. Untuk AWS nilainya, Anda juga dapat

menggunakanhttps://signin.aws.amazon.com/saml, seperti yang ditunjukkan pada contoh berikut.

NameIDelemen dapat memiliki nilai persisten, sementara, atau terdiri dari Format lengkap URI seperti yang disediakan oleh solusi IDP. Nilai persisten menunjukkan bahwa nilai dalam NameID adalah sama untuk pengguna di antara sesi. Jika nilainya bersifat sementara, pengguna memiliki NameID nilai yang berbeda untuk setiap sesi. Interaksi masuk tunggal mendukung jenis pengidentifikasi berikut:

- urn:oasis:names:tc:SAML:2.0:nameid-format:persistent
- urn:oasis:names:tc:SAML:2.0:nameid-format:transient
- urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress
- urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified
- urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName
- urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName
- urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos
- urn:oasis:names:tc:SAML:2.0:nameid-format:entity

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://signin.aws.amazon.com/saml"/>
  </SubjectConfirmation>
</Subject>
```

> **Important**
>
> Kunci saml:aud konteks berasal dari atribut SAML penerima karena SAML setara dengan bidang OIDC audiens, misalnya,accounts.google.com:aud.

PrincipalTagSAMLatribut

(Opsional) Anda dapat menggunakan elemen Attribute dengan atribut Name diatur ke https://aws.amazon.com/SAML/Attributes/PrincipalTag:*{TagKey}*. Elemen ini memungkinkan

Anda untuk meneruskan atribut sebagai tag sesi dalam SAML pernyataan. Untuk informasi lebih lanjut tentang tag sesi, lihat Lulus tag sesi di AWS STS.

Untuk menyampaikan atribut sebagai tag sesi, sertakan elemen `AttributeValue` yang menentukan nilai tag. Misalnya, untuk melewati pasangan nilai kunci tag `Project` = `Marketing` dan `CostCenter` = `12345`, gunakan atribut berikut. Termasuk elemen `Attribute` terpisah untuk setiap tag.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

Untuk mengatur tanda di atas sebagai transitif, sertakan elemen `Attribute` lain dengan atribut `Name` yang diatur ke `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Ini adalah atribut multinilai opsional yang menetapkan tanda sesi Anda sebagai transitif. Tag transitif tetap ada saat Anda menggunakan SAML sesi untuk mengambil peran lain. AWS Ini dikenal sebagai perangkaian peran. Misalnya, untuk mengatur tag `Principal` dan `CostCenter` yang bersifat transitif, gunakan atribut berikut untuk menentukan kunci.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

`Role`SAMLatribut

Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/Role`. Elemen ini berisi satu atau lebih `AttributeValue` elemen yang mencantumkan penyedia IAM identitas dan peran yang pengguna dipetakan oleh idP Anda. Penyedia IAM peran dan IAM identitas ditentukan sebagai pasangan yang dibatasi koma ARNs dalam format yang sama dengan `PrincipalArn` parameter `RoleArn` dan parameter yang diteruskan ke. AssumeRoleWithSAML Elemen ini harus memuat setidaknya satu pasangan peran-penyedia (elemen `AttributeValue`), dan dapat berisi beberapa pasangan. Jika elemen berisi beberapa pasangan, maka pengguna diminta untuk memilih peran mana yang akan diambil ketika mereka menggunakan Web SSO untuk masuk ke AWS Management Console.

> **Important**
>
> Nilai dari atribut `Name` dalam tag `Attribute` peka terhadap huruf besar-kecil. Nilai harus diatur ke `https://aws.amazon.com/SAML/Attributes/Role` dengan tepat.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

`RoleSessionName`SAMLatribut

Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. Elemen ini berisi satu elemen `AttributeValue` yang menyediakan pengenal untuk kredensial sementara yang dikeluarkan ketika peran diambil. Anda dapat menggunakan ini untuk mengasosiasikan kredensial sementara dengan pengguna yang menggunakan aplikasi Anda. Elemen ini digunakan untuk menampilkan informasi pengguna di AWS Management Console. Nilai dalam elemen `AttributeValue` harus terdiri dari 2 hingga 64 karakter, dapat berisi hanya karakter alfanumerik, garis bawah, dan karakter berikut: . , + = @ - (tanda hubung). Tidak dapat berisi spasi. Nilai ini umumnya adalah ID pengguna (`johndoe`) atau alamat email (`johndoe@example.com`). Nilai tidak boleh berupa nilai yang mencakup spasi, seperti nama tampilan pengguna (`John Doe`).

> **Important**
>
> Nilai dari atribut `Name` dalam tag `Attribute` peka terhadap huruf besar-kecil. Nilai harus diatur ke `https://aws.amazon.com/SAML/Attributes/RoleSessionName` dengan tepat.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

SessionDurationSAMLatribut

(Opsional) Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/SessionDuration"`. Elemen ini berisi satu `AttributeValue` elemen yang menentukan berapa lama pengguna dapat mengakses AWS Management Console sebelum harus meminta kredensi sementara baru. Nilai adalah bilangan bulat yang mewakili jumlah detik untuk sesi. Nilai dapat berkisar dari 900 detik (15 menit) hingga 43200 detik (12 jam). Jika atribut ini tidak ada, maka kredensialnya bertahan selama satu jam (nilai default `DurationSeconds` parameter `AssumeRoleWithSAML`API).

Untuk menggunakan atribut ini, Anda harus mengonfigurasi SAML penyedia untuk menyediakan akses masuk tunggal ke titik akhir web login AWS Management Console melalui konsol di. `https://`*`region-code`*`.signin.aws.amazon.com/saml` Untuk daftar kemungkinan *`region-code`* nilai, lihat kolom Region di titik akhir AWS Masuk. Anda dapat menggunakan yang berikut ini secara opsionalURL:`https://signin.aws.amazon.com/static/saml`. Perhatikan bahwa atribut ini hanya memperluas sesi ke AWS Management Console. Atribut ini tidak dapat memperpanjang masa pakai kredensial lainnya. Namun, jika ada dalam `AssumeRoleWithSAML` API panggilan, itu dapat digunakan untuk mempersingkat durasi sesi. Masa pakai kredensial default yang dikembalikan oleh panggilan adalah 60 menit.

Perhatikan juga bahwa jika atribut `SessionNotOnOrAfter` juga ditentukan, nilai yang lebih sedikit dari kedua atribut, `SessionDuration` atau `SessionNotOnOrAfter`, menetapkan durasi maksimum sesi konsol.

Saat Anda mengaktifkan sesi konsol dengan durasi yang diperpanjang, risiko penyusupan kredensial meningkat. Untuk membantu mengurangi risiko ini, Anda dapat langsung menonaktifkan sesi konsol aktif untuk peran apa pun dengan memilih Cabut Sesi pada halaman Ringkasan Peran di konsol. IAM Untuk informasi selengkapnya, lihat Mencabut kredensi IAM keamanan sementara peran.

> **Important**
>
> Nilai dari atribut `Name` dalam tag `Attribute` peka terhadap huruf besar-kecil. Nilai harus diatur ke `https://aws.amazon.com/SAML/Attributes/SessionDuration` dengan tepat.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
```

```
</Attribute>
```

SourceIdentitySAMLatribut

(Opsional) Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Elemen ini berisi satu `AttributeValue` elemen yang menyediakan pengenal untuk orang atau aplikasi yang menggunakan IAM peran. Nilai untuk identitas sumber tetap ada saat Anda menggunakan SAML sesi untuk mengambil peran lain yang AWS dikenal sebagai rantai peran. Nilai untuk identitas sumber hadir dalam permintaan untuk setiap tindakan yang diambil selama sesi peran. Nilai yang ditetapkan tidak dapat diubah selama sesi peran. Administrator kemudian dapat menggunakan AWS CloudTrail log untuk memantau dan mengaudit informasi identitas sumber untuk menentukan siapa yang melakukan tindakan dengan peran bersama.

Nilai dalam elemen `AttributeValue` harus terdiri dari 2 hingga 64 karakter, dapat berisi hanya karakter alfanumerik, garis bawah, dan karakter berikut: . , + = @ - (tanda hubung). Tidak dapat berisi spasi. Nilai biasanya merupakan atribut yang berhubungan dengan pengguna seperti id pengguna (`johndoe`) atau alamat email (`johndoe@example.com`). Nilai tidak boleh berupa nilai yang mencakup spasi, seperti nama tampilan pengguna (`John  Doe`). Untuk informasi selengkapnya tentang penggunaan identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

> **Important**
>
> Jika SAML pernyataan Anda dikonfigurasi untuk menggunakan `SourceIdentity`atribut, maka kebijakan kepercayaan peran Anda juga harus menyertakan `sts:SetSourceIdentity` tindakan, jika tidak, operasi peran asumsi akan gagal. Untuk informasi selengkapnya tentang penggunaan identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Untuk meneruskan atribut identitas sumber, sertakan elemen `AttributeValue` yang menentukan nilai identitas sumber. Misalnya, untuk meneruskan identitas sumber `DiegoRamirez` menggunakan atribut berikut.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>DiegoRamirez</AttributeValue>
```

Memetakan SAML atribut untuk AWS mempercayai kunci konteks kebijakan

Tabel di bagian ini mencantumkan SAML atribut yang umum digunakan dan bagaimana mereka memetakannya untuk mempercayai kunci konteks kondisi kebijakan AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci dengan nilai yang disertakan dalam pernyataan yang menyertai permintaan SAML akses.

> **Important**
>
> Kunci ini hanya tersedia dalam kebijakan IAM kepercayaan (kebijakan yang menentukan siapa yang dapat mengambil peran) dan tidak berlaku untuk kebijakan izin.

Dalam tabel eduPerson dan eduOrg atribut, nilai diketik baik sebagai string atau sebagai daftar string. Untuk nilai string, Anda dapat menguji nilai-nilai ini dalam kebijakan IAM kepercayaan menggunakan `StringEquals` atau `StringLike` kondisi. Untuk nilai yang berisi daftar string, Anda dapat menggunakan `ForAnyValue` dan `ForAllValues` operator set kebijakan untuk menguji nilai-nilai dalam kebijakan kepercayaan.

> **Note**
>
> Anda harus menyertakan hanya satu klaim per kunci AWS konteks. Jika Anda menyertakan lebih dari satu, hanya satu klaim yang akan dipetakan.

Tabel berikut menunjukkan eduPerson dan eduOrg atribut.

eduPerson atau eduOrg atribut (**`Name`**kunci)	Peta ke kunci AWS konteks ini (**`FriendlyName`** kunci)	Tipe
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPersonAffiliation`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	String

eduPerson atau eduOrg atribut (**Name**kunci)	Peta ke kunci AWS konteks ini (**FriendlyName** kunci)	Tipe
urn:oid:1.3.6.1.4.1.5923.1.1.1.4	eduPersonOrgUnitDN	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.1.1.5	eduPersonPrimaryAffiliation	String
urn:oid:1.3.6.1.4.1.5923.1.1.1.6	eduPersonPrincipalName	String
urn:oid:1.3.6.1.4.1.5923.1.1.1.7	eduPersonEntitlement	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.1.1.8	eduPersonPrimaryOrgUnitDN	String
urn:oid:1.3.6.1.4.1.5923.1.1.1.9	eduPersonScopedAffiliation	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.1.1.10	eduPersonTargetedID	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.1.1.11	eduPersonAssurance	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.2.1.2	eduOrgHomePageURI	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.2.1.3	eduOrgIdentityAuthNPolicyURI	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.2.1.4	eduOrgLegalName	Daftar string
urn:oid:1.3.6.1.4.1.5923.1.2.1.5	eduOrgSuperiorURI	Daftar string

eduPerson atau eduOrg atribut (**Name**kunci)	Peta ke kunci AWS konteks ini (**FriendlyName** kunci)	Tipe
`urn:oid:1.3.6.1.4.1.5923.1.2.1.6`	`eduOrgWhitePagesURI`	Daftar string
`urn:oid:2.5.4.3`	`cn`	Daftar string

Tabel berikut menunjukkan atribut Active Directory.

Atribut AD	Peta ke kunci AWS konteks ini	Tipe
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name`	`name`	String
`http://schemas.xmlsoap.org/claims/CommonName`	`commonName`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname`	`givenName`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname`	`surname`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`	`mail`	String
`http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid`	`uid`	String

Tabel berikut menunjukkan X.500 atribut.

atribut X.500	Peta ke kunci AWS konteks ini	Tipe
2.5.4.3	commonName	String
2.5.4.4	surname	String
2.4.5.42	givenName	String
2.5.4.45	x500UniqueIdentifier	String
0.9.2342.19200300100.1.1	uid	String
0.9.2342.19200300100.1.3	mail	String
0.9.2342.19200300.100.1.45	organizationStatus	Tali

Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console

Anda dapat menggunakan peran untuk mengonfigurasi penyedia identitas (IDP) yang sesuai dengan SAMP 2.0 dan mengizinkan pengguna federasi Anda AWS mengakses. AWS Management Console Peran ini memberikan izin kepada pengguna untuk melaksanakan tugas di konsol. Jika Anda ingin memberi pengguna gabungan SAML cara lain untuk mengakses AWS, lihat salah satu topik berikut:

- AWS CLI: Beralih ke IAM peran (AWS CLI)
- Tools for Windows PowerShell: Beralih ke IAM peran (Alat untuk Windows PowerShell)
- AWS API: Beralih ke IAM peran (AWS API)

Gambaran Umum

Diagram berikut menggambarkan alur untuk sistem masuk tunggal yang mendukung SAML.

> **Note**
>
> Penggunaan khusus SAMP ini berbeda dari yang lebih umum diilustrasikan di SAML2.0 federasi karena alur kerja ini membuka AWS Management Console atas nama pengguna.

Ini memerlukan penggunaan titik akhir AWS masuk alih-alih langsung memanggil API. `AssumeRoleWithSAML` Titik akhir memanggil API untuk pengguna dan mengembalikan URL yang secara otomatis mengalihkan peramban pengguna ke AWS Management Console.

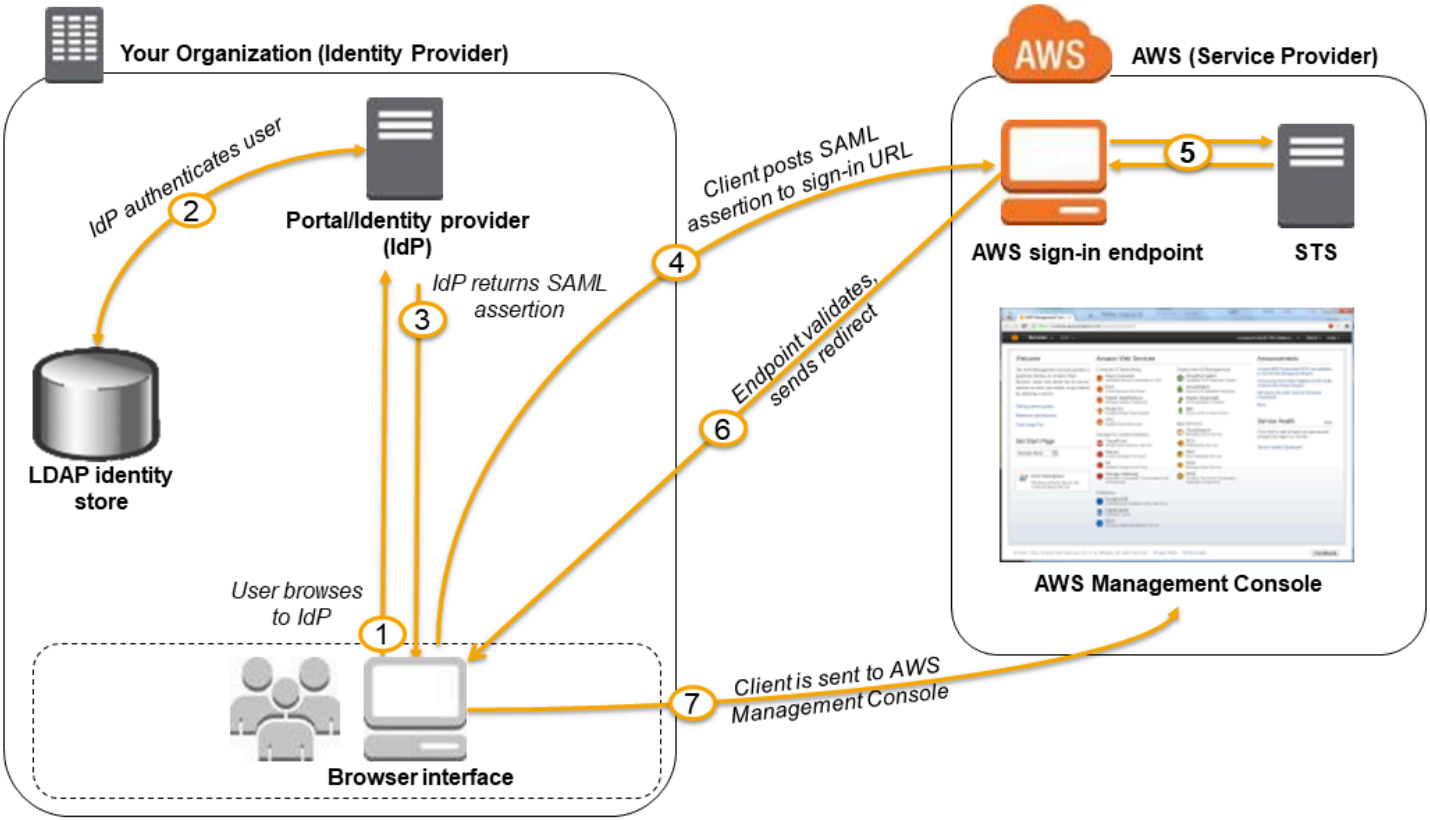


Diagram ini menggambarkan langkah-langkah berikut:

1. Pengguna menelusuri portal organisasi Anda dan memilih opsi untuk membuka AWS Management Console. Dalam organisasi Anda, portal biasanya merupakan fungsi dari IDP Anda yang menangani pertukaran kepercayaan antara organisasi Anda dan organisasi. AWS Misalnya, di Layanan Federasi Active Directory, URL portal adalah: `https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`
2. Portal tersebut memverifikasi identitas pengguna di organisasi Anda.
3. Portal tersebut menghasilkan tanggapan autentikasi SAML yang mencakup pernyataan yang mengidentifikasi pengguna dan menyertakan atribut tentang pengguna. Anda juga dapat mengonfigurasi IdP untuk menyertakan atribut pernyataan SAML yang disebut `SessionDuration` yang menentukan durasi sesi konsol yang valid. Anda juga dapat

mengonfigurasi IdP untuk meneruskan atribut sebagai tanda sesi. Portal mengirimkan respons ini ke browser klien.

4. Browser klien dialihkan ke titik akhir masuk AWS tunggal dan memposting pernyataan SAMP.
5. Titik akhir meminta kredensial keamanan sementara atas nama pengguna dan membuat URL masuk konsol yang menggunakan kredensial tersebut.
6. AWS mengirim URL masuk kembali ke klien sebagai pengalihan.
7. Peramban klien diarahkan ulang ke AWS Management Console. Jika tanggapan autentikasi SAML mencakup atribut yang memetakan ke beberapa peran IAM, pengguna pertama-tama diminta memilih peran untuk mengakses konsol.

Dari sudut pandang pengguna, prosesnya terjadi secara transparan: Pengguna mulai di portal internal organisasi Anda dan berakhir di AWS Management Console, tanpa harus memberikan AWS kredensil apa pun.

Lihat bagian berikut untuk gambaran umum tentang cara mengkonfigurasi perilaku ini bersama dengan tautan ke langkah-langkah rinci.

Konfigurasikan jaringan Anda sebagai penyedia SAMP untuk AWS

Di dalam jaringan organisasi, Anda mengonfigurasi penyimpanan identitas (seperti Windows Active Directory) untuk bekerja dengan IdP berbasis SAML seperti Windows Active Directory Federation Services, Shibboleth, dll. Menggunakan IdP, Anda membuat dokumen metadata yang menjelaskan organisasi Anda sebagai IdP dan menyertakan kunci autentikasi. Anda juga mengonfigurasi portal organisasi Anda untuk merutekan permintaan pengguna AWS Management Console ke titik akhir AWS SAMP untuk otentikasi menggunakan pernyataan SAMP. Cara Anda mengonfigurasi IdP untuk menghasilkan file metadata.xml tergantung pada IdP Anda. Lihat dokumentasi IdP untuk instruksi, atau lihat Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS untuk tautan ke dokumentasi web bagi banyak penyedia SAML yang didukung.

Membuat penyedia SAML di IAM

Selanjutnya, Anda masuk ke AWS Management Console dan pergi ke konsol IAM. Di sana Anda membuat penyedia SAMP baru, yang merupakan entitas di IAM yang menyimpan informasi tentang IDP organisasi Anda. Sebagai bagian dari proses ini, Anda mengunggah dokumen metadata yang dibuat oleh perangkat lunak IdP di organisasi Anda pada bagian sebelumnya. Untuk detailnya, lihat Buat penyedia SAML identitas di IAM.

Konfigurasikan izin AWS untuk pengguna federasi Anda

Langkah selanjutnya adalah menciptakan peran IAM yang membangun hubungan kepercayaan antara IAM dan IDP organisasi Anda. Peran ini harus mengidentifikasi IdP sebagai prinsipal (entitas tepercaya) untuk tujuan federasi. Peran ini juga menentukan apa yang diizinkan dilakukan oleh pengguna yang diautentikasi oleh IDP organisasi Anda. AWS Anda dapat menggunakan konsol IAM untuk membuat peran ini. Saat Anda membuat kebijakan kepercayaan yang menunjukkan siapa yang dapat mengambil peran tersebut, Anda menentukan penyedia SAML yang Anda buat sebelumnya di IAM. Anda juga menentukan satu atau beberapa atribut SAML yang harus dicocokkan pengguna agar diizinkan untuk menjalankan peran tersebut. Misalnya, Anda dapat menyebutkan bahwa hanya pengguna yang nilai `eduPersonOrgDN` SAML-nya adalah `ExampleOrg` yang diperbolehkan untuk masuk. Wizard peran secara otomatis menambahkan ketentuan untuk menguji atribut `saml:aud` guna memastikan bahwa peran tersebut diambil hanya untuk masuk ke AWS Management Console. Kebijakan kepercayaan untuk peran tersebut mungkin akan terlihat seperti ini:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {"StringEquals": {
      "saml:edupersonorgdn": "ExampleOrg",
      "saml:aud": "https://signin.aws.amazon.com/saml"
    }}
  }]
}
```

> **Note**
>
> IDP SAMP yang digunakan dalam kebijakan kepercayaan peran harus berada dalam akun yang sama dengan peran tersebut.

Anda dapat menyertakan titik akhir regional untuk `saml:aud` atribut dihttps://*region-code*`.signin.aws.amazon.com/static/saml-metadata.xml`. Untuk daftar kemungkinan nilai *kode wilayah, lihat kolom Region* di titik akhir AWS Masuk.

Untuk kebijakan izin dalam peran tersebut, Anda menentukan izin seperti yang Anda inginkan untuk peran, pengguna, atau grup. Misalnya, jika pengguna dari organisasi Anda diizinkan untuk mengelola instans Amazon EC2, Anda secara eksplisit mengizinkan tindakan Amazon EC2 dalam kebijakan izin. Anda dapat melakukannya dengan menetapkan kebijakan terkelola, seperti kebijakan terkelola Akses Penuh Amazon EC2.

Untuk detail tentang membuat peran untuk IdP SAML, lihat Buat peran untuk federasi SAML 2.0 (konsol).

Selesaikan konfigurasi dan buat pernyataan SAML

Beri tahu SAMP IDP Anda yang AWS merupakan penyedia layanan Anda dengan menginstal `saml-metadata.xml` file yang ditemukan di atau. `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` `https://signin.aws.amazon.com/static/saml-metadata.xml` Untuk daftar kemungkinan nilai *kode wilayah, lihat kolom Region* di titik akhir AWS Masuk.

Cara Anda menginstal file tergantung pada IdP Anda. Beberapa penyedia memberi Anda opsi untuk mengetik URL, tempat IdP mendapatkan dan menginstal file untuk Anda. Lainnya mengharuskan Anda mengunduh file dari URL lalu menyediakannya sebagai file lokal. Lihat dokumentasi IdP untuk detail, atau lihat Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS untuk tautan ke dokumentasi web bagi banyak penyedia SAML yang didukung.

Anda juga mengonfigurasi informasi yang Anda inginkan untuk diteruskan oleh IdP sebagai atribut SAML ke AWS sebagai bagian dari respons autentikasi. Sebagian besar informasi ini muncul AWS sebagai kunci konteks kondisi yang dapat Anda evaluasi dalam kebijakan Anda. Kunci syarat ini memastikan bahwa hanya pengguna yang diberi otorisasi dalam konteks yang tepat yang diberi izin untuk mengakses sumber daya AWS Anda. Anda dapat menentukan jendela waktu yang dibatasi saat konsol dapat digunakan. Anda juga dapat menentukan waktu maksimum (hingga 12 jam) yang dapat diakses pengguna konsol sebelum harus menyegarkan kembali kredensial mereka. Untuk detailnya, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi.

Lihat SAML respons di browser Anda

Prosedur berikut menjelaskan cara melihat SAML respons dari penyedia layanan Anda dari browser Anda saat memecahkan masalah terkait SAML 2.0.

Untuk semua peramban, kunjungi halaman tempat Anda dapat mereproduksi masalah. Kemudian ikuti langkah-langkah untuk peramban yang sesuai:

Topik

Google Chrome

Untuk melihat SAML respons di Chrome

Langkah-langkah ini diuji menggunakan versi 106.0.5249.103 (Build Resmi) (arm64) dari Google Chrome. Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Tekan F12 untuk memulai konsol Alat Pengembang.
2. Pilih tab Jaringan, lalu pilih Pertahankan log di kiri atas jendela Alat Pengembang.
3. Mereproduksi masalah.
4. (Opsional) Jika kolom Metode tidak terlihat di panel log Jaringan Alat Pengembang, klik kanan pada label kolom mana pun dan pilih Metode untuk menambahkan kolom.
5. Cari SAMLPosting di panel log Jaringan Alat Pengembang. Pilih baris itu, lalu lihat tab Payload di bagian atas. Cari SAMLResponseelemen yang berisi permintaan yang dikodekan. Nilai yang terkait adalah respons yang dienkodekan dengan Base64.

Mozilla Firefox

Untuk melihat SAML respons di Firefox

Prosedur ini diuji pada versi 105.0.3 (64-bit) dari Mozilla Firefox. Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Tekan F12 untuk memulai konsol Alat Pengembang Web.
2. Pilih tab Jaringan.
3. Di kanan atas jendela Alat Pengembang Web, pilih opsi (ikon roda gigi kecil). Pilih Persiste logs.
4. Mereproduksi masalah.
5. (Opsional) Jika kolom Metode tidak terlihat di panel log Jaringan Alat Pengembang Web, klik kanan pada label kolom mana pun dan pilih Metode untuk menambahkan kolom.

6. Cari POSTSAMLdi meja. Pilih baris itu, lalu lihat tab Permintaan dan temukan SAMLResponseelemennya. Nilai yang terkait adalah respons yang dienkodekan dengan Base64.

Apple Safari

Untuk melihat SAML respons di Safari

Langkah-langkah ini diuji menggunakan versi 16.0 (17614.1.25.9.10, 17614) dari Apple Safari. Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Aktifkan Web Inspector di Safari. Buka jendela Preferensi , pilih tab Lanjutan , lalu pilih Tampilkan menu Kembangkan di bilah menu.
2. Sekarang Anda dapat membuka Web Inspector. Pilih Kembangkan di bilah menu, lalu pilih Show Web Inspector.
3. Pilih tab Jaringan.
4. Di kiri atas jendela Web Inspector, pilih opsi (ikon lingkaran kecil yang berisi tiga garis horizontal). Pilih Pertahankan Log.
5. (Opsional) Jika kolom Metode tidak terlihat di panel log Web Inspector Network, klik kanan pada label kolom mana pun dan pilih Metode untuk menambahkan kolom.
6. Mereproduksi masalah.
7. Cari POSTSAMLdi meja. Pilih baris itu, lalu lihat tab Header.
8. Cari SAMLResponseelemen yang berisi permintaan yang dikodekan. Gulir ke bawah untuk menemukan `Request Data` dengan nama `SAMLResponse`. Nilai yang terkait adalah respons yang dienkodekan dengan Base64.

Apa yang harus dilakukan dengan respons yang dikodekan Base64 SAML

Setelah Anda menemukan elemen SAML respons yang disandikan Base64 di browser Anda, salin dan gunakan alat decoding Base-64 favorit Anda untuk mengekstrak respons yang ditandai. XML

> **Tips keamanan**
>
> Karena data SAML respons yang Anda lihat mungkin berisi data keamanan sensitif, sebaiknya Anda tidak menggunakan decoder base64 online. Sebagai gantinya gunakan

alat yang diinstal pada komputer lokal Anda yang tidak mengirim SAML data Anda melalui jaringan.

Opsi bawaan untuk sistem Windows (PowerShell):

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

Opsi bawaan untuk sistem macOS dan Linux:

```
$ echo "base64encodedtext" | base64 --decode
```

Tinjau nilai dalam file yang didekodekan

Tinjau nilai dalam file SAML respons yang diterjemahkan.

- Verifikasi bahwa nilai untuk atribut SAML:nameID cocok dengan nama pengguna untuk pengguna yang diautentikasi.
- Tinjau nilai untuk https://aws.amazon.com/SAML/ Atribut/Peran. SAMLPenyedia ARN dan peka huruf besar/kecil, dan ARNharus cocok dengan sumber daya di akun Anda.
- Tinjau nilai untuk https://aws.amazon.com/SAML/ Atribut/RoleSessionName. Nilai harus sesuai dengan nilai dalam aturan klaim.
- Jika Anda mengonfigurasi nilai atribut untuk alamat email atau nama akun, pastikan nilainya benar. Nilai harus sesuai dengan alamat email atau nama akun pengguna yang diautentikasi.

Periksa kesalahan dan konfirmasikan konfigurasi

Periksa apakah nilai mengandung kesalahan, dan konfirmasikan bahwa konfigurasi berikut sudah benar.

- Aturan klaim memenuhi elemen yang diperlukan dan ARNs semuanya benar. Untuk informasi selengkapnya, lihat Konfigurasikan IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim.
- Anda mengunggah file metadata terbaru dari IDP Anda ke dalam penyedia Anda. AWS SAML Untuk informasi selengkapnya, lihat Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console.

- Anda mengonfigurasi kebijakan kepercayaan IAM peran dengan benar. Untuk informasi selengkapnya, lihat Metode untuk mengambil peran.

Kredensi keamanan sementara di IAM

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan pengguna tepercaya dengan kredenial keamanan sementara yang dapat mengontrol akses ke Anda AWS sumber daya. Kredensil keamanan sementara bekerja hampir identik dengan kredensil kunci akses jangka panjang, dengan perbedaan berikut:

- Kredensial keamanan sementara bersifat jangka pendek, seperti namanya. Konfigurasi dapat berlangsung selama beberapa menit hingga beberapa jam. Setelah kredensialnya kedaluwarsa, AWS tidak lagi mengenali mereka atau mengizinkan segala jenis akses dari API permintaan yang dibuat dengan mereka.
- Kredensial keamanan sementara tidak disimpan dengan pengguna tetapi dihasilkan secara dinamis dan diberikan kepada pengguna saat diminta. Ketika (atau bahkan sebelum) kredensial keamanan sementara kedaluwarsa, pengguna dapat meminta kredensial baru, selama pengguna yang memintanya masih memiliki izin untuk melakukannya.

Akibatnya, kredensil sementara memiliki keunggulan sebagai berikut dibandingkan kredensil jangka panjang:

- Anda tidak perlu mendistribusikan atau menanamkan jangka panjang AWS kredensi keamanan dengan aplikasi.
- Anda dapat memberikan akses ke AWS sumber daya untuk pengguna tanpa harus mendefinisikan AWS identitas untuk mereka. Kredensi sementara adalah dasar untuk peran dan federasi identitas.
- Kredensi keamanan sementara memiliki masa pakai yang terbatas, jadi Anda tidak perlu memperbaruinya atau mencabutnya secara eksplisit saat tidak lagi diperlukan. Setelah kredensial keamanan sementara berakhir, kredensial tersebut tidak dapat digunakan kembali. Anda dapat menentukan berapa lama kredensial berlaku, hingga batas maksimum.

AWS STS and AWS daerah

Kredensi keamanan sementara dihasilkan oleh AWS STS. Secara default, AWS STS adalah layanan global dengan satu titik akhir di`https://sts.amazonaws.com`. Namun, Anda juga dapat memilih untuk membuat AWS STS APIpanggilan ke titik akhir di Wilayah lain yang didukung. Ini dapat

mengurangi latensi (server lag) dengan mengirim permintaan ke server di Wilayah yang secara geografis lebih dekat dengan Anda. Tidak peduli dari Wilayah mana kredensial Anda berasal, mereka bekerja secara global. Untuk informasi selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Skenario umum untuk kredensial sementara

Kredensi sementara berguna dalam skenario yang melibatkan federasi identitas, delegasi, akses lintas akun, dan peran. IAM

Federasi identitas

Anda dapat mengelola identitas pengguna Anda di sistem eksternal di luar AWS dan memberikan pengguna yang masuk dari sistem tersebut akses untuk melakukan AWS tugas dan akses Anda AWS sumber daya. IAMmendukung dua jenis federasi identitas. Dalam kedua kasus tersebut, identitas disimpan di luar AWS. Perbedaannya adalah di mana sistem eksternal berada — di pusat data Anda atau pihak ketiga eksternal di web. Untuk membandingkan fitur kredensil keamanan sementara untuk federasi identitas, lihat. Bandingkan AWS STS kredensialnya

Untuk informasi lebih lanjut tentang penyedia identitas eksternal, lihat Penyedia dan federasi identitas.

- Federasi OpenID Connect (OIDC) - Anda dapat mengizinkan pengguna masuk menggunakan penyedia identitas pihak ketiga yang terkenal seperti Login with Amazon, Facebook, Google, atau penyedia OIDC 2.0 yang kompatibel untuk aplikasi seluler atau web Anda, Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Menggunakan OIDC federasi membantu Anda menjaga Akun AWS aman, karena Anda tidak perlu mendistribusikan kredensil keamanan jangka panjang, seperti kunci akses IAM pengguna, dengan aplikasi Anda. Untuk informasi selengkapnya, lihat OIDCfederasi.

  AWS STS OIDCfederasi mendukung Login with Amazon, Facebook, Google, dan penyedia identitas yang kompatibel dengan OpenID Connect (OIDC).

  > **Note**
  >
  > Untuk aplikasi seluler, kami sarankan Anda menggunakan Amazon Cognito. Anda dapat menggunakan layanan ini dengan AWS SDKsuntuk pengembangan seluler untuk membuat identitas unik bagi pengguna dan mengautentikasi mereka untuk akses aman ke AWS sumber daya. Amazon Cognito mendukung penyedia identitas yang sama dengan AWS

STS, dan juga mendukung akses (tamu) yang tidak diautentikasi dan memungkinkan Anda memigrasi data pengguna saat pengguna masuk. Amazon Cognito juga menyediakan API operasi untuk menyinkronkan data pengguna sehingga dipertahankan saat pengguna berpindah antar perangkat. Untuk informasi selengkapnya, lihat Otentikasi dengan Amplify di Dokumentasi Amplify.

- SAMLfederasi — Anda dapat mengautentikasi pengguna di jaringan organisasi Anda, dan kemudian memberikan akses kepada pengguna tersebut AWS tanpa membuat yang baru AWS identitas untuk mereka dan mengharuskan mereka untuk masuk dengan kredenal masuk yang berbeda. Ini dikenal sebagai pendekatan masuk tunggal untuk akses sementara. AWS STS mendukung standar terbuka seperti Security Assertion Markup Language (SAML) 2.0, yang dengannya Anda dapat menggunakan Microsoft AD FS untuk memanfaatkan Microsoft Active Directory Anda. Anda juga dapat menggunakan SAML 2.0 untuk mengelola solusi Anda sendiri untuk menyatukan identitas pengguna. Untuk informasi selengkapnya, lihat SAML2.0 federasi.
  - Pialang federasi khusus - Anda dapat menggunakan sistem otentikasi organisasi Anda untuk memberikan akses ke AWS sumber daya. Untuk contoh skenario, lihat Aktifkan akses broker identitas khusus ke AWS konsol.
  - Federasi menggunakan SAML 2.0 — Anda dapat menggunakan sistem otentikasi organisasi Anda dan SAML memberikan akses ke AWS sumber daya. Untuk informasi lebih lanjut dan contoh skenario, lihat SAML2.0 federasi.

Peran untuk akses lintas akun

Banyak organisasi mempertahankan lebih dari satu Akun AWS. Menggunakan peran dan akses lintas akun, Anda dapat menentukan identitas pengguna dalam satu akun, dan menggunakan identitas tersebut untuk mengakses AWS sumber daya di akun lain milik organisasi Anda. Ini dikenal sebagai pendekatan pendelegasian ke akses sementara. Untuk informasi selengkapnya tentang cara membuat peran lintas akun, lihat Membuat peran untuk mendelegasikan izin ke pengguna IAM. Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? .

Peran untuk Amazon EC2

Jika Anda menjalankan aplikasi di EC2 instans Amazon dan aplikasi tersebut memerlukan akses ke AWS sumber daya, Anda dapat memberikan kredensil keamanan sementara untuk instans Anda saat Anda meluncurkannya. Kredensial keamanan sementara ini tersedia untuk semua aplikasi yang berjalan pada instans, sehingga Anda tidak perlu menyimpan kredensial jangka panjang apa pun

pada instans. Untuk informasi selengkapnya, lihat Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2.

Untuk mempelajari lebih lanjut tentang EC2 kredensil peran IAM Amazon, lihat IAMperan untuk Amazon EC2 di Panduan Pengguna Amazon Elastic Compute Cloud.

Lainnya AWS layanan

Anda dapat menggunakan kredensi keamanan sementara untuk mengakses sebagian besar AWS layanan. Untuk daftar layanan yang menerima kredensial keamanan sementara, lihat AWS layanan yang bekerja dengan IAM.

Contoh aplikasi yang menggunakan kredensial sementara

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan pengguna tepercaya dengan kredenial keamanan sementara yang dapat mengontrol akses ke Anda AWS sumber daya. Untuk informasi lebih lanjut tentang AWS STS, lihat Kredensi keamanan sementara di IAM. Untuk melihat bagaimana Anda dapat menggunakan AWS STS untuk mengelola kredensil keamanan sementara, Anda dapat mengunduh contoh aplikasi berikut yang menerapkan skenario contoh lengkap:

- Mengaktifkan Federation untuk AWS Menggunakan Windows Active DirectoryADFS,, dan SAML 2.0. Menunjukkan cara menghapus akses menggunakan federasi perusahaan ke AWS menggunakan Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0, dan SAML (Security Assertion Markup Language) 2.0.
- Aktifkan akses broker identitas khusus ke AWS konsol. Menunjukkan cara membuat proxy federasi kustom yang memungkinkan single sign-on (SSO) sehingga pengguna Active Directory yang ada dapat masuk ke AWS Management Console.
- Cara Menggunakan Shibboleth untuk Single Sign-On ke AWS Management Console. . Menunjukkan cara menggunakan Shibboleth dan SAMLmemberi pengguna akses masuk tunggal () ke SSO AWS Management Console.

Sampel untuk OIDC federasi

Contoh aplikasi berikut menggambarkan cara menggunakan OIDCfederation dengan penyedia seperti Login with Amazon, Amazon Cognito, Facebook, atau Google. Anda dapat menukar otentikasi dari penyedia ini untuk sementara AWS kredensi keamanan untuk mengakses AWS layanan.

- Tutorial Amazon Cognito - Kami menyarankan Anda menggunakan Amazon Cognito dengan AWS SDKsuntuk pengembangan mobile. Amazon Cognito adalah cara paling mudah untuk mengelola identitas untuk aplikasi seluler, serta menyediakan fitur tambahan seperti sinkronisasi dan identitas lintas perangkat. Untuk informasi selengkapnya tentang Amazon Cognito, lihat Otentikasi dengan Amplify di Dokumentasi Amplify.

Sumber daya tambahan untuk kredensial keamanan sementara

Skenario dan aplikasi berikut dapat memandu Anda dalam menggunakan kredensial keamanan sementara:

- Bagaimana cara mengintegrasikan AWS STS SourceIdentity dengan penyedia identitas Anda. Posting ini menunjukkan kepada Anda cara mengatur AWS STS `SourceIdentity`atribut saat menggunakan Okta, Ping, atau OneLogin sebagai idP Anda.
- OIDCfederasi. Bagian ini membahas cara mengonfigurasi IAM peran saat Anda menggunakan OIDC federasi dan. `AssumeRoleWithWebIdentity` API
- APIAkses aman dengan MFA. Topik ini menjelaskan cara menggunakan peran untuk memerlukan otentikasi multi-faktor (MFA) untuk melindungi API tindakan sensitif di akun Anda.

Untuk informasi selengkapnya tentang kebijakan dan izin di AWS lihat topik-topik berikut:

- Manajemen akses untuk AWS sumber daya
- Logika evaluasi kebijakan.
- Mengelola Izin Akses ke Sumber Daya Amazon S3 Anda di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.
- Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? .

Bandingkan AWS STS kredensialnya

Tabel berikut membandingkan fitur API operasi AWS STS yang mengembalikan kredensi keamanan sementara. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk meminta kredensial keamanan sementara dengan mengambil peran, lihat Metode untuk mengambil peran. Untuk mempelajari tentang berbagai AWS STS API operasi yang memungkinkan Anda melewati tag sesi, lihatLulus tag sesi di AWS STS.

> **Note**
>
> Anda dapat mengirim AWS STS API panggilan ke titik akhir global atau ke salah satu titik akhir Regional. Jika Anda memilih titik akhir yang lebih dekat dengan Anda, Anda dapat mengurangi latensi dan meningkatkan kinerja panggilan AndaAPI. Anda juga dapat memilih untuk mengarahkan panggilan Anda ke titik akhir Regional alternatif jika Anda tidak lagi dapat berkomunikasi dengan titik akhir awal. Jika Anda menggunakan salah satu dari berbagai AWS SDKs, maka gunakan SDK metode itu untuk menentukan Wilayah sebelum Anda melakukan API panggilan. Jika Anda membuat HTTP API permintaan secara manual, maka Anda harus mengarahkan permintaan ke titik akhir yang benar sendiri. Untuk informasi lebih lanjut, lihat bagian AWS STS dari Wilayah dan Titik Akhir dan Kelola AWS STS dalam sebuah Wilayah AWS.

AWS STS API	Siapa yang bisa menelepon	Masa pakai kredensi (min | maks | default)	MFAdukuran [1]	Dukungan kebijakan sesi[2]	Pembatasan pada kredensi sementara yang dihasilkan
AssumeRe	IAMpengguna atau IAM peran dengan kredensi keamanan sementara yang ada	15 m | Pengaturan durasi sesi maksimum | 1 jam	Ya	Ya	Tidak bisa menelepon `GetFederationToken` atau `GetSessionToken` .
AssumeReWithSAM	Setiap pengguna; penelepon harus melewati respons SAML otentikasi yang menunjukkan otentikasi dari penyedia identitas yang dikenal	15 m | Pengaturan durasi sesi maksimum | 1 jam	Tidak	Ya	Tidak bisa menelepon `GetFederationToken` atau `GetSessionToken` .

AWS STS API	Siapa yang bisa menelepon	Masa pakai kredensi (min | maks | default)	MFAduku an [1]	Dukungar kebijakan sesi[2]	Pembatasan pada kredensi sementara yang dihasilkan
AssumeR eWithWeb dentity	Setiap pengguna; penelepon harus melewati JWT token yang OIDC sesuai yang menunjukkan otentikasi dari penyedia identitas yang dikenal	15 m | Pengatura n durasi sesi maksimur | 1 jam	Tidak	Ya	Tidak bisa menelepon `GetFederationToken` atau `GetSessionToken` .
GetFedera tionToken	IAMpengguna atau Pengguna root akun AWS	IAMpengg na: 15 m | 36 jam | 12 jam Pengguna root: 15 m | 1 jam | 1 jam	Tidak	Ya	Tidak dapat memanggil IAM operasi menggunakan AWS CLI atau AWS API. Batasan ini tidak berlaku untuk sesi konsol. Tidak dapat memanggil AWS STS operasi kecuali `GetCallerIdentity` .1 SSOke konsol diizinkan.1

AWS STS API	Siapa yang bisa menelepon	Masa pakai kredensi (min | maks | default)	MFAdukuı an [1]	Dukungar kebijakan sesi[2]	Pembatasan pada kredensi sementara yang dihasilkan
GetSessic nToken	IAMpengguna atau Pengguna root akun AWS	IAMpengg na: 15 m | 36 jam | 12 jam Penggunа root: 15 m | 1 jam | 1 jam	Ya	Tidak	Tidak dapat memanggil IAM API operasi kecuali MFA informasi disertakan dengan permintaan. Tidak dapat memanggil AWS STS API operasi kecuali `AssumeRole` atau`GetCallerIdentity` . SSOke konsol tidak diizinkan .1

[1] MFAdukungan. Anda dapat menyertakan informasi tentang perangkat otentikasi multi-faktor (MFA) saat Anda memanggil AssumeRole dan GetSessionToken API operasi. Ini memastikan bahwa kredenal keamanan sementara yang dihasilkan dari API panggilan hanya dapat digunakan oleh pengguna yang diautentikasi dengan perangkat. MFA Untuk informasi selengkapnya, lihat APIAkses aman dengan MFA.

[2] Dukungan kebijakan sesi. Kebijakan sesi adalah kebijakan yang Anda jalankan sebagai parameter saat Anda secara terprogram membuat sesi sementara untuk peran atau pengguna federasi. Kebijakan ini membatasi izin dari peran atau kebijakan berbasis identitas pengguna yang ditetapkan untuk sesi. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas entitas dan kebijakan sesi. Kebijakan sesi tidak dapat digunakan untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari peran yang sedang diasumsikan. Untuk informasi lebih lanjut tentang izin sesi peran, lihat Kebijakan sesi.

[3] Pengaturan durasi sesi maksimum. Gunakan parameter `DurationSeconds` untuk menentukan durasi sesi peran Anda dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk

peran tersebut. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

GetCallerIdentity. Tidak ada izin yang diperlukan untuk melakukan operasi ini. Jika administrator menambahkan kebijakan ke IAM pengguna atau peran Anda yang secara eksplisit menolak akses ke `sts:GetCallerIdentity` tindakan, Anda masih dapat melakukan operasi ini. Izin tidak diperlukan karena informasi yang sama dikembalikan ketika IAM pengguna atau peran ditolak aksesnya. Untuk melihat contoh tanggapan, lihat Saya tidak berwenang untuk melakukan: iam: DeleteVirtual MFADevice.

Single sign-on (SSO) ke konsol. Untuk mendukungSSO, Anda AWS dapat memanggil federasi endpoint (`https://signin.aws.amazon.com/federation`) dan meneruskan kredensi keamanan sementara. Titik akhir mengembalikan token yang dapat Anda gunakan untuk membuat tanda tangan URL pengguna langsung ke konsol tanpa memerlukan kata sandi. Untuk informasi selengkapnya, lihat Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console dan Cara Mengaktifkan Akses Lintas Akun ke Konsol AWS Manajemen di Blog AWS Keamanan.

[6] Setelah Anda mendapatkan kredensial sementara Anda, Anda tidak dapat mengakses Konsol Manajemen AWS Management Console dengan meneruskan kredensial ke titik akhir masuk tunggal gabungan. Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

Token pembawa layanan

Beberapa AWS layanan mengharuskan Anda memiliki izin untuk mendapatkan token pembawa AWS STS layanan sebelum Anda dapat mengakses sumber daya mereka secara terprogram. Layanan ini mendukung protokol yang mengharuskan Anda untuk menggunakan token pembawa alih-alih menggunakan tradisionalAWS Signature Version 4 untuk API permintaan. Ketika Anda melakukan AWS CLI atau AWS API operasi yang memerlukan token pembawa, AWS layanan meminta token pembawa atas nama Anda. Layanan memberi Anda token, yang kemudian dapat Anda gunakan untuk melakukan operasi selanjutnya dalam layanan tersebut.

AWS STS token pembawa layanan mencakup informasi dari otentikasi utama asli Anda yang dapat memengaruhi izin Anda. Informasi ini dapat mencakup tanda utama, tanda sesi, dan kebijakan sesi. Access key ID token dimulai dengan awalan `ABIA`. Ini membantu Anda mengidentifikasi operasi yang dilakukan menggunakan token pembawa layanan di CloudTrail log Anda.

> **Important**
>
> Token pembawa dapat digunakan hanya untuk panggilan ke layanan yang membuatnya dan di Wilayah tempat pembuatannya. Anda tidak dapat menggunakan token pembawa untuk melakukan operasi di layanan atau Wilayah lain.

Contoh layanan yang mendukung token pembawa adalah AWS CodeArtifact. Sebelum Anda dapat berinteraksi dengan AWS CodeArtifact menggunakan manajer paket sepertiNPM, Maven, atauPIP, Anda harus memanggil operasi. `aws codeartifact get-authorization-token` Operasi ini mengembalikan token pembawa yang dapat Anda gunakan untuk melakukan AWS CodeArtifact operasi. Atau, Anda dapat menggunakan perintah `aws codeartifact login` yang menyelesaikan operasi yang sama dan kemudian mengonfigurasi klien Anda secara otomatis.

Jika Anda melakukan tindakan dalam AWS layanan yang menghasilkan token pembawa untuk Anda, Anda harus memiliki izin berikut dalam kebijakan AndaIAM:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

Untuk contoh token pembawa layanan, lihat Menggunakan kebijakan berbasis identitas untuk AWS CodeArtifact dalam panduan pengguna. AWS CodeArtifact

Minta kredensi keamanan sementara

Untuk meminta kredensi keamanan sementara, Anda dapat menggunakan AWS Security Token Service (AWS STS) operasi di. AWS API Ini termasuk operasi untuk membuat dan menyediakan kredensi keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke sumber daya Anda AWS . Untuk informasi lebih lanjut tentang AWS STS, lihatKredensi keamanan sementara di IAM. Untuk mempelajari lebih lanjut tentang berbagai metode berbeda yang dapat

Anda gunakan untuk meminta kredensial keamanan sementara dengan mengasumsikan peran, lihat Metode untuk mengambil peran.

Untuk memanggil API operasi, Anda dapat menggunakan salah satu dari AWS SDKs. SDKsTersedia untuk berbagai bahasa pemrograman dan lingkungan, termasuk Java,. NET, Python, Ruby, Android, dan iOS. SDKsMenangani tugas-tugas seperti menandatangani permintaan Anda secara kriptografis, mencoba ulang permintaan jika perlu, dan menangani respons kesalahan. Anda juga dapat menggunakan AWS STS QueryAPI, yang dijelaskan dalam AWS Security Token Service APIReferensi. Akhirnya, dua alat baris perintah mendukung AWS STS perintah: AWS Command Line Interface, dan AWS Tools for Windows PowerShell.

AWS STS APIOperasi membuat sesi baru dengan kredensi keamanan sementara yang mencakup access key pair dan token sesi. Pasangan pasangan access key terdiri dari access key ID dan secret access key. Pengguna (atau aplikasi yang dijalankan pengguna) dapat menggunakan kredensial ini untuk mengakses sumber daya Anda. Anda dapat membuat sesi peran dan meneruskan kebijakan sesi dan tag sesi secara terprogram menggunakan AWS STS API operasi. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas peran dan kebijakan sesi. Untuk informasi lebih lanjut tentang kebijakan sesi, lihat Kebijakan sesi. Untuk informasi lebih lanjut tentang tanda sesi, lihat Lulus tag sesi di AWS STS.

> **Note**
>
> Ukuran token sesi yang dikembalikan AWS STS API operasi tidak tetap. Kami sangat menyarankan agar Anda tidak membuat asumsi tentang ukuran maksimum. Ukuran token umumnya kurang dari 4096 byte, tetapi itu dapat bervariasi.

Menggunakan AWS STS dengan AWS Wilayah

Anda dapat mengirim AWS STS API panggilan ke titik akhir global atau ke salah satu titik akhir Regional. Jika Anda memilih titik akhir yang lebih dekat dengan Anda, Anda dapat mengurangi latensi dan meningkatkan kinerja panggilan AndaAPI. Anda juga dapat memilih untuk mengarahkan panggilan Anda ke titik akhir Regional alternatif jika Anda tidak lagi dapat berkomunikasi dengan titik akhir awal. Jika Anda menggunakan salah satu dari berbagai AWS SDKs, maka gunakan SDK metode itu untuk menentukan Wilayah sebelum Anda melakukan API panggilan. Jika Anda membuat HTTP API permintaan secara manual, maka Anda harus mengarahkan permintaan ke titik akhir yang benar sendiri. Untuk informasi lebih lanjut, lihat bagian AWS STS dari Wilayah dan Titik Akhir dan Kelola AWS STS dalam sebuah Wilayah AWS.

Berikut ini adalah API operasi yang dapat Anda gunakan untuk memperoleh kredensil sementara untuk digunakan di AWS lingkungan dan aplikasi Anda.

Meminta kredensi untuk delegasi dan federasi lintas akun melalui pialang identitas khusus

`AssumeRole`APIOperasi ini berguna untuk memungkinkan IAM pengguna yang ada mengakses AWS sumber daya yang belum mereka akses. Misalnya, pengguna mungkin memerlukan akses ke sumber daya di tempat lain Akun AWS. Hal ini juga berguna sebagai sarana untuk sementara mendapatkan akses istimewa — misalnya, untuk menyediakan otentikasi multi-faktor (). MFA Anda harus memanggil ini API menggunakan kredensi aktif. Untuk mempelajari siapa yang dapat memanggil operasi ini, lihatBandingkan AWS STS kredensialnya. Untuk informasi selengkapnya, silakan lihat Membuat peran untuk mendelegasikan izin ke pengguna IAM dan APIAkses aman dengan MFA.

Untuk meminta kredensi keamanan sementara untuk delegasi lintas akun dan federasi melalui broker identitas khusus

1. Otentikasi dengan kredensi AWS keamanan Anda. Panggilan ini harus dilakukan dengan menggunakan kredensial keamanan AWS yang berlaku.
2. Panggil operasi `AssumeRole`.

Contoh berikut menunjukkan permintaan sampel dan respons menggunakan `AssumeRole`. Contoh permintaan ini mengasumsikan peran `demo` untuk durasi yang ditentukan dengan disertakan Kebijakan sesi, tanda sesi, ID eksternal, dan identitas sumber. Sesi yang dihasilkan diberi nama `John-session`.

Example Contoh permintaan

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws::iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A
%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*%22%2C
%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
```

```
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

Nilai kebijakan yang ditunjukkan pada contoh sebelumnya adalah versi URL -encoded dari kebijakan berikut:

```
{"Version":"2012-10-17","Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

Parameter AUTHPARAMS dalam contoh adalah placeholder untuk tanda tangan Anda. Tanda tangan adalah informasi otentikasi yang harus Anda sertakan dengan AWS HTTP API permintaan. Sebaiknya gunakan AWS SDKsuntuk membuat API permintaan, dan satu manfaat melakukannya adalah penandatanganan permintaan penandatanganan untuk Anda. SDKs Jika Anda harus membuat dan menandatangani API permintaan secara manual, lihat Menandatangani AWS Permintaan Dengan Menggunakan Tanda Tangan Versi 4 di bagian Referensi Umum Amazon Web Servicesuntuk mempelajari cara menandatangani permintaan.

Selain kredensil keamanan sementara, respons tersebut mencakup Amazon Resource Name (ARN) untuk pengguna federasi dan waktu kedaluwarsa kredensialnya.

Example Contoh tanggapan

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
```

```
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>8</PackedPolicySize>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> AWS Konversi memampatkan kebijakan sesi dan tag sesi yang diteruskan ke dalam format biner yang dikemas yang memiliki batas terpisah. Permintaan Anda bisa gagal untuk batas ini bahkan jika teks biasa Anda memenuhi persyaratan lain. Elemen respons `PackedPolicySize` menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas.

Meminta kredensi melalui penyedia OIDC

`AssumeRoleWithWebIdentity`APIOperasi mengembalikan satu set kredensi keamanan sementara untuk pengguna federasi yang diautentikasi melalui penyedia identitas publik. Contoh penyedia identitas publik termasuk Login with Amazon, Facebook, Google, atau penyedia identitas yang kompatibel dengan OpenID Connect (OIDC). Menggunakan operasi ini berarti bahwa pengguna Anda tidak memerlukan IAM identitas AWS atau identitas mereka sendiri. Untuk informasi selengkapnya, lihat OIDCfederasi.

> **Note**
>
> Alih-alih menelepon langsung`AssumeRoleWithWebIdentity`, kami menyarankan Anda menggunakan Amazon Cognito dan penyedia kredensi Amazon Cognito dengan for mobile development. AWS SDKs Untuk informasi selengkapnya, lihat Otentikasi dengan Amplify di Dokumentasi Amplify.

Jika Anda tidak menggunakan Amazon Cognito, Anda menyebut `AssumeRoleWithWebIdentity` tindakan. AWS STS

1. Panggil operasi `AssumeRoleWithWebIdentity`.

   Ini adalah panggilan yang tidak ditandatangani, artinya Anda tidak perlu mengautentikasi kredensi AWS keamanan sebelum membuat permintaan.

   > **Note**
   >
   > Sebuah panggilan ke `AssumeRoleWithWebIdentity` tidak ditandatangani (dienkripsi). Oleh karena itu, Anda hanya boleh menyertakan kebijakan sesi opsional jika permintaan dikirim melalui perantara terpercaya. Dalam hal ini, seseorang dapat mengubah kebijakan untuk menghapus pembatasan.

2. Saat Anda menelepon `AssumeRoleWithWebIdentity` AWS , verifikasi keaslian token. Misalnya, tergantung pada penyedia, AWS mungkin melakukan panggilan ke penyedia dan menyertakan token yang telah dilewati aplikasi. Dengan asumsi bahwa penyedia identitas memvalidasi token, AWS mengembalikan informasi berikut kepada Anda:

   - Serangkaian kredensial keamanan sementara. Ini terdiri dari access key ID, secret access key, dan token sesi.
   - ID peran dan peran ARN yang diasumsikan.
   - Suatu nilai `SubjectFromWebIdentityToken` yang memuat ID pengguna unik.

3. Gunakan kredenal keamanan sementara yang dikembalikan dalam respons untuk melakukan AWS API panggilan. Ini adalah proses yang sama dengan melakukan AWS API panggilan dengan kredensi keamanan jangka panjang. Perbedaannya adalah Anda harus menyertakan token sesi, yang memungkinkan AWS verifikasi bahwa kredensil keamanan sementara valid.

Aplikasi Anda harus menyimpan kredensial. Sebagaimana diketahui, secara default kredensial kedaluwarsa setelah satu jam. Jika Anda tidak menggunakan operasi mazonSTSCredentialsPenyedia A di AWS SDK, terserah Anda dan aplikasi Anda untuk menelepon `AssumeRoleWithWebIdentity` lagi. Panggil operasi ini untuk mendapatkan rangkaian kredensial keamanan sementara yang baru sebelum kredensial lama kedaluwarsa.

Meminta kredensi melalui penyedia identitas 2.0 SAML

`AssumeRoleWithSAML`APIOperasi mengembalikan satu set kredensi keamanan sementara untuk pengguna federasi yang diautentikasi oleh sistem identitas organisasi Anda yang ada. Pengguna juga harus menggunakan SAML2.0 (Security Assertion Markup Language) untuk

meneruskan informasi otentikasi dan otorisasi ke. AWS APIOperasi ini berguna dalam organisasi yang telah mengintegrasikan sistem identitas mereka (seperti Windows Active Directory atau OpenLDAP) dengan perangkat lunak yang dapat menghasilkan SAML pernyataan. Integrasi tersebut menyediakan informasi tentang identitas dan izin pengguna (seperti ctive Directory Federation Services atau Shibboleth). Untuk informasi selengkapnya, lihat SAML2.0 federasi.

1. Panggil operasi `AssumeRoleWithSAML`.

   Ini adalah panggilan yang tidak ditandatangani, artinya Anda tidak perlu mengautentikasi kredensi AWS keamanan sebelum membuat permintaan.

   > **Note**
   >
   > Sebuah panggilan ke `AssumeRoleWithSAML` tidak ditandatangani (dienkripsi). Oleh karena itu, Anda hanya boleh menyertakan kebijakan sesi opsional jika permintaan dikirim melalui perantara terpercaya. Dalam hal ini, seseorang dapat mengubah kebijakan untuk menghapus pembatasan.

2. Saat Anda menelepon `AssumeRoleWithSAML` AWS , verifikasi keaslian pernyataan. SAML Dengan asumsi bahwa penyedia identitas memvalidasi pernyataan, AWS mengembalikan informasi berikut kepada Anda:

   - Serangkaian kredensial keamanan sementara. Ini terdiri dari access key ID, secret access key, dan token sesi.
   - ID peran dan peran ARN yang diasumsikan.
   - `Audience`Nilai yang berisi nilai `Recipient` atribut `SubjectConfirmationData` elemen SAML pernyataan.
   - `Issuer`Nilai yang berisi nilai `Issuer` elemen SAML pernyataan.
   - `NameQualifier`Elemen yang berisi nilai hash yang dibangun dari `Issuer` nilai, Akun AWS ID, dan nama ramah SAML penyedia. Saat digabungkan dengan elemen `Subject`, mereka dapat mengidentifikasi pengguna federasi secara unik.
   - `Subject`Elemen yang berisi nilai `NameID` elemen dalam `Subject` elemen SAML pernyataan.
   - Suatu elemen `SubjectType` yang menunjukkan format elemen `Subject`. Nilainya bisa`persistent`,`transient`, atau penuh `Format` URI dari `NameID` elemen `Subject` dan yang digunakan dalam SAML pernyataan Anda. Untuk informasi tentang elemen `NameID` `Format` atribut, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi.

3. Gunakan kredenal keamanan sementara yang dikembalikan dalam respons untuk melakukan AWS API panggilan. Ini adalah proses yang sama dengan melakukan AWS API panggilan dengan kredensi keamanan jangka panjang. Perbedaannya adalah bahwa Anda harus menyertakan token sesi, yang memungkinkan AWS memverifikasi bahwa kredensial keamanan sementara valid.

Aplikasi Anda harus menyimpan kredensial. Secara default kredensial kedaluwarsa setelah satu jam. Jika Anda tidak menggunakan tindakan mazonSTSCredentialsPenyedia A di dalamnya AWS SDK, terserah Anda dan aplikasi Anda untuk menelepon `AssumeRoleWithSAML` lagi. Panggil operasi ini untuk mendapatkan rangkaian kredensial keamanan sementara yang baru sebelum kredensial lama kedaluwarsa.

Meminta kredensi melalui pialang identitas khusus

`GetFederationToken`APIOperasi mengembalikan satu set kredensi keamanan sementara untuk pengguna federasi. Ini API berbeda dari `AssumeRole` periode kedaluwarsa default jauh lebih lama (12 jam, bukan satu jam). Sebagai tambahan, Anda dapat menggunakan parameter `DurationSeconds` untuk menentukan durasi untuk kredensial keamanan sementara agar tetap valid. Kredensi yang dihasilkan berlaku untuk durasi yang ditentukan, antara 900 detik (15 menit) hingga 129.600 detik (36 jam). Masa kedaluwarsa yang lebih lama dapat membantu mengurangi jumlah panggilan AWS karena Anda tidak perlu mendapatkan kredensi baru sesering mungkin.

1. Otentikasi dengan kredensi AWS keamanan pengguna spesifik Anda. IAM Panggilan ini harus dilakukan menggunakan kredensi AWS keamanan yang valid.
2. Panggil operasi `GetFederationToken`.

`GetFederationToken`Panggilan mengembalikan kredensil keamanan sementara yang terdiri dari token sesi, kunci akses, kunci rahasia, dan kedaluwarsa. Anda dapat menggunakan `GetFederationToken` jika anda ingin mengelola izin di dalam organization anda (misalnya, menggunakan aplikasi proksi untuk menetapkan izin).

Contoh berikut menunjukkan permintaan sampel dan respons yang menggunakan `GetFederationToken`. Permintaan contoh ini menggabungkan pengguna pemanggil untuk durasi yang ditentukan dengan kebijakan sesi ARN dan tag sesi. Sesi yang dihasilkan diberi nama `Jane-session`.

Example Contoh permintaan

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

Kebijakan yang ARN ditunjukkan pada contoh sebelumnya mencakup -encoded berikut ini: URL ARN

`arn:aws:iam::123456789012:policy/Role1policy`

Selain itu, perhatikan bahwa parameter &AUTHPARAMS dalam contoh dimaksudkan sebagai kerangka untuk informasi autentikasi. Ini adalah tanda tangan, yang harus Anda sertakan dengan AWS HTTP API permintaan. Sebaiknya gunakan AWS SDKsuntuk membuat API permintaan, dan satu manfaat melakukannya adalah penandatanganan permintaan penandatanganan untuk Anda. SDKs Jika Anda harus membuat dan menandatangani API permintaan secara manual, buka AWS Permintaan Menandatangani Dengan Menggunakan Tanda Tangan Versi 4 di bagian Referensi Umum Amazon Web Servicesuntuk mempelajari cara menandatangani permintaan.

Selain kredensil keamanan sementara, respons tersebut mencakup Amazon Resource Name (ARN) untuk pengguna federasi dan waktu kedaluwarsa kredensialnya.

Example Contoh tanggapan

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
```

```
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>4</PackedPolicySize>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> AWS Konversi memampatkan kebijakan sesi dan tag sesi yang diteruskan ke dalam format biner yang dikemas yang memiliki batas terpisah. Permintaan Anda bisa gagal untuk batas ini bahkan jika teks biasa Anda memenuhi persyaratan lain. Elemen respons `PackedPolicySize` menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas.

AWS menyarankan agar Anda memberikan izin di tingkat sumber daya (misalnya, Anda melampirkan kebijakan berbasis sumber daya ke bucket Amazon S3), Anda dapat menghilangkan parameternya. `Policy` Namun, jika Anda tidak menyertakan kebijakan untuk pengguna federasi, kredensial keamanan sementara tidak akan memberikan izin apa pun. Dalam kasus ini, Anda harus menggunakan kebijakan sumber daya untuk memberikan akses pengguna gabungan ke sumber daya AWS Anda.

Misalnya, anggaplah Akun AWS nomor Anda 111122223333, dan Anda memiliki bucket Amazon S3 yang ingin Anda izinkan untuk diakses oleh Susan. Kredensial keamanan sementara Susan tidak menyertakan kebijakan untuk bucket. Dalam hal ini, Anda perlu memastikan bahwa ember memiliki kebijakan dengan ARN yang cocok dengan SusanARN, seperti`arn:aws:sts::111122223333:federated-user/Susan`.

Meminta kredensi untuk pengguna di lingkungan yang tidak tepercaya

`GetSessionToken`APIOperasi mengembalikan satu set kredensi keamanan sementara ke pengguna yang sudah adaIAM. Ini berguna untuk memberikan keamanan yang ditingkatkan, seperti mengizinkan AWS permintaan hanya ketika MFA diaktifkan untuk IAM pengguna. Karena kredensialnya bersifat sementara, mereka memberikan keamanan yang ditingkatkan ketika Anda memiliki IAM pengguna yang mengakses sumber daya Anda melalui lingkungan yang kurang aman. Contoh lingkungan yang kurang aman termasuk perangkat seluler atau peramban web.

1. Otentikasi dengan kredensi AWS keamanan pengguna spesifik Anda. IAM Panggilan ini harus dilakukan menggunakan kredensi AWS keamanan yang valid.
2. Panggil operasi `GetSessionToken`.
3. `GetSessionToken`mengembalikan kredensi keamanan sementara yang terdiri dari token sesi, ID kunci akses, dan kunci akses rahasia.

Secara default, kredensi keamanan sementara untuk IAM pengguna berlaku selama maksimal 12 jam. Tetapi Anda dapat meminta durasi sesingkat 15 menit atau selama 36 jam menggunakan parameter `DurationSeconds`. Untuk alasan keamanan, token untuk sebuah Pengguna root akun AWS dibatasi hingga durasi satu jam.

Contoh berikut menunjukkan permintaan sampel dan respons menggunakan `GetSessionToken`. Jawaban juga mencakup waktu kedaluwarsa kredensial keamanan sementara.

Example Contoh Permintaan

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
&AUTHPARAMS
```

Parameter `AUTHPARAMS` dalam contoh adalah placeholder untuk tanda tangan Anda. Tanda tangan adalah informasi otentikasi yang harus Anda sertakan dengan AWS HTTP API permintaan. Sebaiknya gunakan AWS SDKsuntuk membuat API permintaan, dan satu manfaat melakukannya adalah penandatanganan permintaan penandatanganan untuk Anda. SDKs Jika Anda harus membuat dan menandatangani API permintaan secara manual, buka AWS Permintaan Menandatangani Dengan Menggunakan Tanda Tangan Versi 4 di bagian Referensi Umum Amazon Web Servicesuntuk mempelajari cara menandatangani permintaan.

Example Contoh tanggapan

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

Secara opsional, `GetSessionToken` permintaan dapat menyertakan `SerialNumber` dan `TokenCode` nilai untuk verifikasi otentikasi AWS multi-faktor ()MFA. Jika nilai yang diberikan valid, AWS STS berikan kredensil keamanan sementara yang mencakup status otentikasi. MFA Kredensi keamanan sementara kemudian dapat digunakan untuk mengakses API operasi atau AWS situs web MFA yang dilindungi selama MFA otentikasi valid.

Contoh berikut menunjukkan `GetSessionToken` permintaan yang menyertakan kode MFA verifikasi dan nomor seri perangkat.

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> Panggilan untuk AWS STS dapat ke titik akhir global atau ke salah satu titik akhir Regional yang Anda aktifkan. Akun AWS Untuk informasi lebih lanjut, lihat bagian AWS STS pada Wilayah dan Titik Akhir.
> Parameter `AUTHPARAMS` dalam contoh adalah placeholder untuk tanda tangan Anda. Tanda tangan adalah informasi otentikasi yang harus Anda sertakan dengan AWS HTTP API permintaan. Sebaiknya gunakan AWS SDKsuntuk membuat API permintaan, dan satu manfaat melakukannya adalah penandatanganan permintaan penandatanganan untuk Anda. SDKs Jika Anda harus membuat dan menandatangani API permintaan secara manual, lihat Menandatangani AWS Permintaan Dengan Menggunakan Tanda Tangan Versi 4 di bagian Referensi Umum Amazon Web Servicesuntuk mempelajari cara menandatangani permintaan.

Gunakan kredensi sementara dengan sumber daya AWS

Anda dapat menggunakan kredensi keamanan sementara untuk membuat permintaan terprogram untuk AWS sumber daya menggunakan AWS CLI atau AWS API (menggunakan). AWS SDKs Kredensi sementara memberikan izin yang sama dengan kredenal keamanan jangka panjang, seperti kredensial pengguna. IAM Namun, ada beberapa perbedaan:

- Saat Anda melakukan panggilan menggunakan kredensi keamanan sementara, panggilan harus menyertakan token sesi, yang dikembalikan bersama dengan kredensial sementara tersebut. AWS menggunakan token sesi untuk memvalidasi kredenal keamanan sementara.
- Kredensi sementara kedaluwarsa setelah interval yang ditentukan. Setelah kredensi sementara kedaluwarsa, panggilan apa pun yang Anda lakukan dengan kredensi tersebut akan gagal, jadi Anda harus membuat kumpulan kredensi sementara yang baru. Kredensi sementara tidak dapat diperpanjang atau disegarkan di luar interval yang ditentukan asli.
- Saat Anda menggunakan kredensial sementara untuk membuat permintaan, prinsipal Anda mungkin mencakup satu set tanda. Tanda ini berasal dari tanda sesi dan tanda yang dilampirkan pada peran yang Anda asumsikan. Untuk informasi lebih lanjut tentang tanda sesi, lihat Lulus tag sesi di AWS STS.

Jika Anda menggunakan, AWS Command Line Interface(AWS CLI) AWS SDKs, atau Tools untuk Windows PowerShell, cara untuk mendapatkan dan menggunakan kredenal keamanan sementara

berbeda dengan konteksnya. Jika Anda menjalankan kode, AWS CLI, atau PowerShell perintah Alat untuk Windows di dalam sebuah EC2 instance, Anda dapat memanfaatkan peran untuk AmazonEC2. Jika tidak, Anda dapat memanggil AWS STS APIuntuk mendapatkan kredensi sementara, dan kemudian menggunakannya secara eksplisit untuk melakukan panggilan ke layanan. AWS

> **Note**
>
> Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan kredenal keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke sumber daya Anda AWS . Untuk informasi lebih lanjut tentang AWS STS, lihatKredensi keamanan sementara di IAM. AWS STS adalah layanan global yang memiliki titik akhir default di`https://sts.amazonaws.com`. Titik akhir ini berada di Wilayah AS Timur (Virginia N.), meskipun kredensi yang Anda dapatkan dari titik ini dan titik akhir lainnya valid secara global. Kredensial ini bekerja dengan layanan dan sumber daya di Wilayah mana pun. Anda juga dapat memilih untuk melakukan AWS STS API panggilan ke titik akhir di salah satu Wilayah yang didukung. Hal ini dapat mengurangi latensi dengan membuat permintaan dari server di Wilayah yang secara geografis lebih dekat dengan Anda. Tidak peduli dari Wilayah mana kredensial Anda berasal, mereka bekerja secara global. Untuk informasi selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Daftar Isi

Menggunakan kredensi sementara di instans Amazon EC2

Jika Anda ingin menjalankan AWS CLI perintah atau kode di dalam EC2 instance, cara yang disarankan untuk mendapatkan kredensi adalah dengan menggunakan peran untuk Amazon. EC2 Anda membuat IAM peran yang menentukan izin yang ingin Anda berikan kepada aplikasi yang berjalan pada EC2 instance. Saat Anda meluncurkan instans, Anda mengaitkan peran dengan instans tersebut.

Aplikasi, AWS CLI, dan Alat untuk PowerShell perintah Windows yang berjalan pada instance kemudian bisa mendapatkan kredenal keamanan sementara otomatis dari metadata instance. Anda tidak perlu secara eksplisit mendapatkan kredensial kerahasiaan keamanan sementara. The AWS SDKs, AWS CLI, and Tools untuk Windows PowerShell secara otomatis mendapatkan kredensialnya dari EC2 Instance Metadata Service (IMDS) dan menggunakannya. Kredensial sementara memiliki izin yang Anda tentukan untuk peran yang terkait dengan instans.

Untuk informasi selengkapnya dan untuk contoh, lihat berikut ini:

- Menggunakan IAM Peran untuk Memberikan Akses ke AWS Sumber Daya di Amazon Elastic Compute Cloud — AWS SDK for Java
- Memberikan Akses Menggunakan IAM Peran — AWS SDK for .NET
- Menciptakan Peran - AWS SDK for Ruby

Menggunakan kredensi keamanan sementara dengan AWS SDKs

Untuk menggunakan kredensi keamanan sementara dalam kode, Anda secara terprogram memanggil AWS STS API like `AssumeRole` dan mengekstrak kredenal dan token sesi yang dihasilkan. Anda kemudian menggunakan nilai-nilai tersebut sebagai kredensi untuk panggilan berikutnya ke. AWS Contoh berikut menunjukkan pseudocode untuk cara menggunakan kredenal keamanan sementara jika Anda menggunakan: AWS SDK

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

Untuk contoh yang ditulis dalam Python (menggunakan AWS SDK for Python (Boto)), lihatBeralih ke IAM peran (AWS API). Contoh ini menunjukkan cara melakukan panggilan `AssumeRole` untuk mendapatkan kredensial keamanan sementara dan kemudian menggunakan kredensial tersebut untuk melakukan panggilan ke Amazon S3

Untuk detail tentang cara menelepon `AssumeRoleGetFederationToken`, dan API operasi lainnya, lihat AWS Security Token Service APIReferensi. Untuk informasi tentang mendapatkan kredensi keamanan sementara dan token sesi dari hasilnya, lihat dokumentasi SDK yang sedang Anda

kerjakan. Anda dapat menemukan dokumentasi untuk semua AWS SDKs pada halaman AWS dokumentasi utama, di bagian SDKsdan Toolkit.

Anda harus memastikan bahwa Anda mendapatkan set kredensial baru sebelum kredensial yang lama kedaluwarsa. Di beberapaSDKs, Anda dapat menggunakan penyedia yang mengelola proses penyegaran kredensi untuk Anda; periksa dokumentasi untuk yang SDK Anda gunakan.

Menggunakan kredensial keamanan sementara dengan AWS CLI

Anda dapat menggunakan kredensial keamanan sementara dengan AWS CLI Ini dapat berguna untuk menguji kebijakan.

Menggunakan AWS CLI, Anda dapat memanggil AWS STS APIsuka `AssumeRole` atau `GetFederationToken` dan kemudian menangkap output yang dihasilkan. Contoh berikut ini menunjukkan panggilan ke `AssumeRole` yang mengirimkan output ke file. Dalam contoh, `profile` parameter diasumsikan sebagai profil dalam file AWS CLI konfigurasi. Hal ini juga diasumsikan untuk referensi kredensi untuk IAM pengguna yang memiliki izin untuk mengambil peran.

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

Setelah perintah selesai, Anda dapat mengekstrak access key ID, secret access key, dan token sesi dari mana pun Anda merutekannya. Anda dapat melakukannya secara manual atau dengan menggunakan skrip. Kemudian Anda dapat menetapkan nilai-nilai ini ke variabel lingkungan.

Ketika Anda menjalankan AWS CLI perintah, AWS CLI mencari kredensi dalam urutan tertentu—pertama dalam variabel lingkungan dan kemudian di file konfigurasi. Oleh karena itu, setelah Anda memasukkan kredensi sementara ke dalam variabel lingkungan, kredensialnya akan AWS CLI digunakan secara default. (Jika Anda menentukan `profile` parameter dalam perintah, AWS CLI melewatkan variabel lingkungan. Sebagai gantinya, AWS CLI tampilan dalam file konfigurasi, yang memungkinkan Anda mengganti kredensi dalam variabel lingkungan jika perlu.)

Contoh berikut menunjukkan bagaimana Anda dapat mengatur variabel lingkungan untuk kredenal keamanan sementara dan kemudian memanggil perintah AWS CLI . Karena tidak ada `profile` parameter yang disertakan dalam AWS CLI perintah, AWS CLI mencari kredensi terlebih dahulu dalam variabel lingkungan dan oleh karena itu menggunakan kredenal sementara.

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
```

```
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Menggunakan kredensi keamanan sementara dengan operasi API

Jika Anda membuat HTTPS API permintaan langsung AWS, Anda dapat menandatangani permintaan tersebut dengan kredenal keamanan sementara yang Anda dapatkan dari AWS Security Token Service ()AWS STS. Untuk melakukan ini, Anda menggunakan ID kunci akses dan kunci akses rahasia yang Anda terima AWS STS. Anda menggunakan access key ID dan secret access key dengan cara yang sama seperti Anda menggunakan kredensial jangka panjang untuk menandatangani permintaan. Anda juga menambahkan token sesi yang Anda terima ke API permintaan Anda AWS STS. Anda menambahkan token sesi ke HTTP header atau ke parameter string kueri bernama`X-Amz-Security-Token`. Anda menambahkan token sesi ke HTTP header atau parameter string kueri, tetapi tidak keduanya. Untuk informasi selengkapnya tentang penandatanganan HTTPS API permintaan, lihat Menandatangani AWS API Permintaan di Referensi Umum AWS.

Informasi lain

Untuk informasi selengkapnya tentang penggunaan AWS STS dengan AWS layanan lain, lihat tautan berikut:

- Amazon S3. Lihat Membuat permintaan menggunakan kredensi sementara IAM pengguna atau Membuat permintaan menggunakan kredensi sementara pengguna gabungan di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.
- Amazon SNS. Lihat Menggunakan kebijakan berbasis identitas dengan Amazon SNS di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.
- Amazon SQS. Lihat Manajemen identitas dan akses di Amazon SQS di Panduan Pengembang Layanan Antrian Sederhana Amazon.
- Amazon SimpleDB. Lihat Menggunakan Kredenal Keamanan Sementara di Panduan Pengembang Amazon SimpleDB.

Izin untuk kredensial keamanan sementara

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan kredenal keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke sumber daya Anda AWS . Untuk informasi lebih lanjut tentang AWS STS, lihatKredensi keamanan sementara di IAM. Setelah AWS STS mengeluarkan kredensial keamanan sementara, kredensial tersebut berlaku selama periode kedaluwarsa dan tidak dapat dicabut. Namun, izin yang diberikan untuk kredensial keamanan sementara dievaluasi setiap kali sebuah permintaan dibuat ketika menggunakan kredensial tersebut, sehingga Anda dapat memperoleh dampak dari mencabut kredensial dengan mengubah hak akses mereka setelah diterbitkan.

Topik berikut menganggap Anda memiliki pengetahuan tentang AWS izin dan kebijakan. Untuk informasi lebih lanjut pada topik ini, lihat Manajemen akses untuk AWS sumber daya.

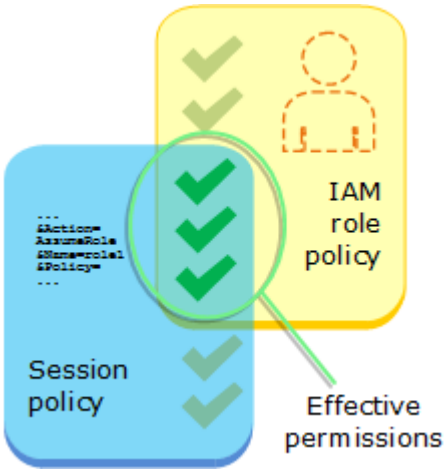
Topik

Izin untuk AssumeRole, AssumeRoleWith SALL, dan AssumeRoleWithWebIdentity

Kebijakan izin dari peran yang sedang diasumsikan menentukan izin untuk kredensial keamanan sementara yang dikembalikan oleh `AssumeRole`, `AssumeRoleWithSAML`, dan `AssumeRoleWithWebIdentity`. Anda menentukan izin ini saat membuat atau memperbarui peran.

Atau, Anda dapat meneruskan kebijakan sesi inline atau terkelola sebagai parameter Operasi API `AssumeRole`, `AssumeRoleWithSAML`, atau `AssumeRoleWithWebIdentity`. Kebijakan sesi membatasi izin untuk sesi kredensial sementara peran tersebut. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas peran dan kebijakan sesi. Anda dapat menggunakan kredensi sementara peran dalam panggilan AWS API berikutnya untuk mengakses sumber daya di akun yang memiliki peran tersebut. Anda tidak dapat menggunakan kebijakan sesi untuk memberikan

lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari peran yang sedang diasumsikan. Untuk mempelajari lebih lanjut tentang bagaimana AWS menentukan izin yang efektif dari suatu peran, lihat Logika evaluasi kebijakan.



Kebijakan yang dilampirkan pada kredensil yang membuat panggilan asli tidak `AssumeRole` dievaluasi oleh AWS saat membuat keputusan otorisasi "izinkan" atau "tolak". Pengguna untuk sementara menyerahkan izin asli yang mendukung izin yang ditetapkan oleh peran yang diasumsikan. Dalam kasus operasi `AssumeRoleWithSAML` dan `AssumeRoleWithWebIdentity` API, tidak ada kebijakan untuk dievaluasi karena pemanggil API bukan AWS identitas.

Contoh: Menetapkan izin menggunakan AssumeRole

Anda dapat menggunakan Operasi API `AssumeRole` dengan berbagai jenis kebijakan. Berikut adalah beberapa contoh.

Kebijakan izin peran

Dalam contoh ini, Anda memanggil operasi API `AssumeRole` tanpa menetapkan kebijakan sesi pada opsional parameter `Policy`. Izin yang diberikan untuk kredensial sementara ditentukan oleh kebijakan izin dari peran yang diasumsikan. Contoh kebijakan izin berikut memberikan izin peran untuk mencantumkan semua objek yang terkandung dalam bucket S3 dengan nama `productionapp`. Hal ini juga memungkinkan peran untuk mendapatkan, menempatkan, dan menghapus objek dalam bucket itu.

Example Contoh kebijakan izin peran

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Kebijakan sesi diberikan sebagai parameter

Bayangkan Anda ingin mengizinkan pengguna untuk mengambil peran yang sama seperti dalam contoh sebelumnya. Tetapi dalam hal ini Anda ingin sesi peran hanya memiliki izin untuk mendapatkan dan memasukkan objek ke dalam bucket S3 `productionapp`. Anda tidak ingin mengizinkan mereka untuk menghapus objek. Salah satu cara untuk mencapainya adalah dengan membuat peran baru dan menentukan izin yang diinginkan dalam kebijakan izin peran tersebut. Cara lain untuk mencapai ini adalah dengan memanggil API `AssumeRole` dan menyertakan kebijakan sesi dalam opsional parameter `Policy` sebagai bagian dari operasi API. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas peran dan kebijakan sesi. Kebijakan sesi tidak dapat digunakan untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari peran yang sedang diasumsikan. Untuk informasi lebih lanjut tentang izin sesi peran, lihat Kebijakan sesi.

Setelah Anda menerima kredensial sementara sesi baru, Anda dapat memberikannya kepada pengguna yang Anda ingin untuk memiliki izin tersebut.

Misalnya, bayangkan kebijakan berikut ini diberikan sebagai parameter panggilan API. Orang yang menggunakan sesi memiliki izin untuk hanya melakukan tindakan berikut:

- Mencantumkan semua objek ke dalam bucket `productionapp`.
- Mendapatkan dan memasukkan objek ke dalam bucket `productionapp`.

Dalam kebijakan sesi berikut, izin `s3:DeleteObject` difilter dan sesi yang diasumsikan tidak diberikan izin `s3:DeleteObject`. Kebijakan menetapkan izin maksimum untuk sesi peran sehingga menggantikan kebijakan izin yang ada pada peran tersebut.

Example Contoh kebijakan sesi melewati Panggilan API **`AssumeRole`**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Kebijakan berbasis sumber daya

Beberapa AWS sumber daya mendukung kebijakan berbasis sumber daya, dan kebijakan ini menyediakan mekanisme lain untuk menentukan izin yang memengaruhi kredensi keamanan sementara. Hanya beberapa sumber daya, seperti bucket Amazon S3, topik Amazon SNS, dan antrean Amazon SQS mendukung kebijakan berbasis sumber daya. Contoh berikut memperluas contoh sebelumnya, menggunakan bucket S3 bernama `productionapp`. Kebijakan berikut ini terlampir di bucket.

Ketika Anda melampirkan kebijakan berbasis sumber daya berikut ini ke bucket `productionapp`, semua pengguna tidak diberi izin untuk menghapus objek dari bucket. (Lihat elemen `Principal` dalam kebijakan.) Ini termasuk semua pengguna peran yang diasumsikan, meskipun kebijakan izin peran memberikan izin `DeleteObject`. Sebuah pernyataan `Deny` yang jelas selalu lebih diutamakan daripada pernyataan `Allow`.

Example Contoh kebijakan bucket

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

Untuk informasi selengkapnya tentang bagaimana beberapa jenis kebijakan digabungkan dan dievaluasi oleh AWS, lihatLogika evaluasi kebijakan.

Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan

IAMPeran adalah objek IAM yang diberi izin. Saat Anda mengambil peran tersebut menggunakan IAM identitas atau identitas dari luar AWS, Anda menerima sesi dengan izin yang ditetapkan ke peran tersebut.

Saat Anda melakukan tindakan AWS, informasi tentang sesi Anda dapat dicatat AWS CloudTrail untuk dipantau oleh administrator akun Anda. Administrator dapat mengkonfigurasi peran untuk meminta identitas untuk melewati string kustom yang mengidentifikasi orang atau aplikasi yang melakukan tindakan di AWS. Informasi identitas ini disimpan sebagai identitas sumber di AWS CloudTrail. Saat administrator meninjau aktivitas CloudTrail, mereka dapat melihat informasi identitas sumber untuk menentukan siapa atau tindakan apa yang dilakukan dengan sesi peran yang diasumsikan.

Setelah identitas sumber ditetapkan, identitas tersebut hadir dalam permintaan untuk AWS tindakan apa pun yang diambil selama sesi peran. Nilai yang ditetapkan tetap ada ketika peran digunakan untuk mengambil peran lain melalui AWS CLI atau AWS API, yang dikenal sebagai rantai peran. Nilai yang ditetapkan tidak dapat diubah selama sesi peran. Administrator dapat mengonfigurasi izin granular berdasarkan keberadaan atau nilai identitas sumber untuk mengontrol AWS tindakan lebih lanjut yang diambil dengan peran bersama. Anda dapat memutuskan apakah atribut identitas sumber dapat digunakan, apakah itu diperlukan, dan apakah nilai dapat digunakan.

Cara Anda menggunakan identitas sumber berbeda dari nama sesi peran dan tanda sesi dalam cara yang penting. Nilai identitas sumber tidak dapat diubah setelah ditetapkan, dan tetap ada untuk

tindakan tambahan yang diambil dengan sesi peran. Berikut cara menggunakan tanda sesi dan nama sesi peran:

- Tag sesi — Anda dapat meneruskan tag sesi saat Anda mengambil peran atau menyatukan pengguna. Tanda sesi hadir ketika peran diasumsikan. Kemudian, Anda dapat menentukan kebijakan yang menggunakan kunci syarat tanda untuk memberikan izin kepada penanggung jawab Anda berdasarkan tanda mereka. Kemudian Anda dapat menggunakan CloudTrail untuk melihat permintaan yang dibuat untuk mengambil peran atau pengguna federasi. Untuk mempelajari lebih lanjut tentang tanda sesi, lihat Lulus tag sesi di AWS STS.
- Nama sesi peran — Anda dapat menggunakan kunci `sts:RoleSessionName` kondisi dalam kebijakan kepercayaan peran untuk mengharuskan pengguna Anda memberikan nama sesi tertentu saat mereka mengambil peran. Nama sesi peran dapat digunakan untuk membedakan sesi peran ketika peran digunakan oleh penanggung jawab yang berbeda. Untuk mempelajari lebih lanjut tentang nama sesi peran, lihat sts: RoleSessionName.

Kami menyarankan Anda menggunakan identitas sumber ketika Anda ingin mengontrol identitas yang mengasumsikan peran. Identitas sumber juga berguna untuk penambangan CloudTrail log untuk menentukan siapa yang menggunakan peran untuk melakukan tindakan.

Topik

- Menyiapkan untuk menggunakan identitas sumber
- Hal yang perlu diketahui tentang identitas sumber
- Izin yang diperlukan untuk menetapkan identitas sumber
- Menentukan identitas sumber ketika mengasumsikan peran
- Menggunakan identitas sumber dengan AssumeRole
- Menggunakan identitas sumber dengan AssumeRoleWith SAML
- Menggunakan identitas sumber dengan AssumeRoleWithWebIdentity
- Mengontrol akses menggunakan informasi identitas sumber
- Melihat identitas sumber di CloudTrail

Menyiapkan untuk menggunakan identitas sumber

Cara Anda mengatur untuk menggunakan identitas sumber tergantung metode yang digunakan ketika peran Anda diasumsikan. Misalnya, IAM pengguna Anda mungkin mengambil peran secara langsung menggunakan `AssumeRole` operasi. Jika Anda memiliki identitas perusahaan, juga dikenal

sebagai identitas tenaga kerja, mereka dapat mengakses sumber daya Anda AWS menggunakan. `AssumeRoleWithSAML` Jika pengguna akhir mengakses aplikasi seluler atau web Anda, mereka mungkin melakukannya menggunakan `AssumeRoleWithWebIdentity`. Berikut ini adalah gambaran umum alur kerja tingkat tinggi untuk membantu Anda memahami bagaimana Anda dapat mengatur untuk memanfaatkan informasi identitas sumber di lingkungan yang ada.

1. Konfigurasikan pengguna dan peran pengujian — Menggunakan lingkungan praproduksi, konfigurasikan pengguna dan peran pengujian, serta konfigurasikan kebijakan mereka untuk memungkinkan pengaturan identitas sumber.

   Jika Anda menggunakan penyedia identitas (IdP) untuk identitas federasi Anda, konfigurasikan IdP Anda untuk melewati atribut pengguna pilihan Anda untuk identitas sumber dalam pernyataan atau token.
2. Asumsikan peran — Uji asumsi peran dan meneruskan identitas sumber dengan pengguna dan peran yang Anda siapkan untuk pengujian.
3. Tinjauan CloudTrail — Tinjau informasi identitas sumber untuk peran pengujian Anda di CloudTrail log Anda.
4. Latih pengguna Anda — Setelah Anda menguji di lingkungan praproduksi, pastikan bahwa pengguna Anda tahu cara meneruskan informasi identitas sumber, jika perlu. Tetapkan tenggat waktu untuk kapan Anda akan meminta pengguna Anda untuk memberikan identitas sumber di lingkungan produksi Anda.
5. Konfigurasikan kebijakan produksi — Konfigurasikan kebijakan Anda untuk lingkungan produksi, lalu tambahkan ke pengguna dan peran produksi Anda.
6. Pantau aktivitas — Pantau aktivitas peran produksi Anda menggunakan CloudTrail log.

Hal yang perlu diketahui tentang identitas sumber

Ingatlah hal-hal berikut ini saat bekerja dengan identitas sumber.

- Kebijakan kepercayaan untuk semua peran yang terhubung ke penyedia identitas (IdP) harus memiliki izin `sts:SetSourceIdentity`. Untuk peran yang tidak memiliki izin ini dalam kebijakan kepercayaan peran, operasi `AssumeRole*` akan gagal. Jika Anda tidak ingin memperbarui kebijakan kepercayaan peran untuk setiap peran, Anda dapat menggunakan instans IdP terpisah untuk meneruskan identitas sumber. Lalu tambahkan izin `sts:SetSourceIdentity` hanya untuk peran yang terhubung ke IdP yang terpisah.

- Ketika identitas menetapkan identitas sumber, kunci `sts:SourceIdentity` terdapat dalam permintaan. Untuk tindakan selanjutnya yang diambil selama sesi peran, kunci `aws:SourceIdentity` terdapat dalam permintaan. AWS tidak mengontrol nilai identitas sumber di salah satu kunci `sts:SourceIdentity` atau `aws:SourceIdentity`. Jika Anda memilih untuk meminta identitas sumber, Anda harus memilih atribut yang Anda inginkan pengguna atau IdP untuk menyediakan. Untuk tujuan keamanan, Anda harus memastikan bahwa Anda dapat mengontrol bagaimana nilai-nilai tersebut disediakan.
- Nilai identitas sumber harus antara 2 dan 64 karakter panjang, dapat berisi hanya karakter alfanumerik, garis bawah, dan karakter berikut:., + = @ - (tanda hubung). Anda tidak dapat menggunakan nilai yang dimulai dengan teks **`aws:`**. Awalan ini dicadangkan untuk penggunaan AWS internal.
- Informasi identitas sumber tidak ditangkap oleh CloudTrail ketika AWS layanan atau peran terkait layanan melakukan tindakan atas nama identitas federasi atau tenaga kerja.

> **Important**
>
> Anda tidak dapat beralih ke peran AWS Management Console yang memerlukan identitas sumber untuk disetel saat peran diasumsikan. Untuk mengambil peran seperti itu, Anda dapat menggunakan AWS CLI atau AWS API untuk memanggil `AssumeRole` operasi dan menentukan parameter identitas sumber.

Izin yang diperlukan untuk menetapkan identitas sumber

Selain tindakan yang cocok dengan API operasi, Anda harus memiliki tindakan khusus izin berikut dalam kebijakan Anda:

```
sts:SetSourceIdentity
```

- Untuk menentukan identitas sumber, prinsipal (IAMpengguna dan peran) harus memiliki izin untuk. `sts:SetSourceIdentity` Sebagai administrator, Anda dapat mengonfigurasi ini dalam kebijakan kepercayaan peran dan kebijakan izin penanggung jawab.
- Ketika Anda mengambil peran dengan peran lain, disebut perangkaian peran, izin untuk `sts:SetSourceIdentity` diperlukan dalam kebijakan perizinan penanggung jawab yang mengasumsikan peran dan dalam kebijakan kepercayaan peran pada peran target. Jika tidak, peran operasi asumsi akan gagal.

- Saat menggunakan identitas sumber, kebijakan kepercayaan peran untuk semua peran yang terhubung ke IdP harus memiliki izin `sts:SetSourceIdentity`. Operasi `AssumeRole*` akan gagal untuk peran apa pun yang terhubung ke IdP tanpa izin ini. Jika Anda tidak ingin memperbarui kebijakan kepercayaan peran untuk setiap peran, Anda dapat menggunakan instans IdP terpisah untuk meneruskan identitas sumber dan menambahkan izin `sts:SetSourceIdentity` hanya untuk peran yang terhubung ke IdP yang terpisah.
- Untuk menetapkan identitas sumber di batas akun, Anda harus menyertakan izin `sts:SetSourceIdentity` di dua tempat. Harus berada dalam kebijakan izin penanggung jawab di akun asal dan kebijakan kepercayaan peran dalam akun target. Anda mungkin perlu melakukannya, misalnya, ketika peran digunakan untuk mengambil peran di akun lain dengan perangkaian peran.

Sebagai administrator akun, bayangkan Anda ingin mengizinkan IAM pengguna `DevUser` di akun Anda untuk mengasumsikan akun yang sama. `Developer_Role` Tetapi Anda ingin mengizinkan tindakan ini hanya jika pengguna telah mengatur identitas sumber ke nama IAM pengguna mereka. Anda dapat melampirkan kebijakan berikut kepada IAM pengguna.

Example Contoh kebijakan berbasis identitas yang dilampirkan DevUser

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
    },
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
```

```
}
```

Untuk memberlakukan nilai identitas sumber yang dapat diterima, Anda dapat mengonfigurasi kebijakan kepercayaan peran berikut. Kebijakan ini memberikan `DevUser` izin IAM pengguna untuk mengambil peran dan menetapkan identitas sumber. Kunci syarat `sts:SourceIdentity` mendefinisikan nilai identitas sumber yang dapat diterima.

Example Contoh kebijakan kepercayaan peran untuk identitas sumber

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
        }
      }
    }
  ]
}
```

Menggunakan kredensil untuk IAM penggunaDevUser, pengguna mencoba untuk mengasumsikan `DeveloperRole` menggunakan permintaan berikut AWS CLI .

Example AssumeRole CLIPermintaan contoh

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

Saat AWS mengevaluasi permintaan, konteks permintaan berisi `sts:SourceIdentity` dari. `DevUser`

Menentukan identitas sumber ketika mengasumsikan peran

Anda dapat menentukan identitas sumber saat menggunakan salah satu AWS STS `AssumeRole*` API operasi untuk mendapatkan kredensil keamanan sementara untuk suatu peran. APIOperasi yang Anda gunakan berbeda tergantung pada kasus penggunaan Anda. Misalnya, jika Anda menggunakan IAM peran untuk memberi IAM pengguna akses ke AWS sumber daya yang biasanya tidak dapat mereka akses, Anda dapat menggunakan `AssumeRole` operasi tersebut. Jika Anda menggunakan federasi identitas perusahaan untuk mengelola pengguna tenaga kerja Anda, Anda dapat menggunakan operasi `AssumeRoleWithSAML`. Jika Anda menggunakan OIDC federasi untuk mengizinkan pengguna akhir mengakses aplikasi seluler atau web Anda, Anda dapat menggunakan `AssumeRoleWithWebIdentity` operasi tersebut. Bagian berikut menjelaskan cara menggunakan identitas sumber dengan setiap operasi. Untuk mempelajari selengkapnya tentang skenario umum untuk kredensial sementara, lihat Skenario umum untuk kredensial sementara.

Menggunakan identitas sumber dengan AssumeRole

`AssumeRole`Operasi mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Anda dapat menggunakan kredensi IAM pengguna atau peran untuk menelepon. `AssumeRole` Untuk meneruskan identitas sumber sambil mengambil peran, gunakan `--source-identity` AWS CLI opsi atau `SourceIdentity` AWS API parameter. Contoh berikut menunjukkan cara menentukan identitas sumber menggunakan AWS CLI.

Example AssumeRole CLIPermintaan contoh

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
--role-session-name Audit \
--source-identity Admin \
```

Menggunakan identitas sumber dengan AssumeRoleWith SAML

Kepala sekolah yang memanggil `AssumeRoleWithSAML` operasi diautentikasi menggunakan federasi SAML berbasis. Operasi ini mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang penggunaan federasi SAML berbasis untuk AWS Management Console akses, lihatMengaktifkan pengguna

federasi SAMP 2.0 untuk mengakses AWS Management Console. Untuk detail tentang AWS CLI atau AWS API akses, lihatSAML2.0 federasi. Untuk tutorial menyiapkan SAML federasi untuk pengguna Active Directory Anda, lihat AWS Otentikasi Federasi dengan Layanan Federasi Direktori Aktif (ADFS) di Blog AWS Keamanan.

Sebagai administrator, Anda dapat mengizinkan anggota direktori perusahaan Anda untuk bergabung AWS menggunakan AWS STS `AssumeRoleWithSAML` operasi. Untuk melakukannya, Anda harus menyelesaikan tugas berikut:

1. Konfigurasikan SAML penyedia di organisasi Anda.
2. Buat SAML penyedia di IAM.
3. Konfigurasikan peran dan izinnya AWS untuk pengguna federasi Anda.
4. Selesai mengkonfigurasi SAML IDP dan membuat pernyataan untuk respon otentikasi. SAML

Untuk menetapkan SAML atribut untuk identitas sumber, sertakan `Attribute` elemen dengan `Name` atribut yang disetel ke`https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Gunakan elemen `AttributeValue` untuk menentukan nilai identitas sumber. Misalnya, anggap Anda ingin meneruskan atribut identitas berikut sebagai identitas sumber.

```
SourceIdentity:DiegoRamirez
```

Untuk meneruskan atribut ini, sertakan elemen berikut dalam SAML pernyataan Anda.

Example Contoh cuplikan pernyataan SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

Menggunakan identitas sumber dengan AssumeRoleWithWebIdentity

Pemanggilan utama `AssumeRoleWithWebIdentity` operasi diautentikasi menggunakan federasi yang sesuai dengan OpenID OIDC Connect (). Operasi ini menghasilkan serangkaian kredensial sementara yang dapat Anda gunakan untuk mengakses sumber daya AWS . Untuk informasi selengkapnya tentang penggunaan OIDC federasi untuk AWS Management Console akses, lihatOIDCfederasi.

Untuk meneruskan identitas sumber dari OpenID Connect (OIDC), Anda harus menyertakan identitas sumber di JSON Web Token ()JWT. Sertakan identitas sumber dalam namespace `https://`

aws.amazon.com/ source_identity dalam token ketika Anda mengajukan permintaan AssumeRoleWithWebIdentity. Untuk mempelajari lebih lanjut tentang OIDC token dan klaim, lihat Menggunakan Token dengan Kumpulan Pengguna di Panduan Amazon Cognito Pengembang.

Misalnya, decoded berikut JWT adalah token yang digunakan untuk memanggil AssumeRoleWithWebIdentity dengan identitas Admin sumber.

Example Contoh Token Web yang didekodekan JSON

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

Mengontrol akses menggunakan informasi identitas sumber

Ketika identitas sumber awalnya ditetapkan, SourceIdentity kunci sts: ada dalam permintaan. Setelah identitas sumber disetel, SourceIdentity kunci aws: hadir di semua permintaan berikutnya yang dibuat selama sesi peran. Sebagai administrator, Anda dapat menulis kebijakan yang memberikan otorisasi bersyarat untuk melakukan AWS tindakan berdasarkan keberadaan atau nilai atribut identitas sumber.

Bayangkan Anda ingin meminta pengembang Anda untuk menetapkan identitas sumber untuk mengambil peran penting yang memiliki izin untuk menulis ke AWS sumber daya penting produksi. Juga bayangkan bahwa Anda memberikan AWS akses ke identitas tenaga kerja Anda menggunakan. AssumeRoleWithSAML Anda hanya ingin developer senior Saanvi dan Diego memiliki akses ke peran, sehingga Anda membuat kebijakan kepercayaan berikut untuk peran tersebut.

Example Contoh kebijakan kepercayaan peran untuk identitas sumber (SAML)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
          "SAML:aud": "https://signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

Kebijakan kepercayaan berisi syarat untuk `sts:SourceIdentity` itu membutuhkan identitas sumber Saanvi atau Diego untuk mengambil peran penting.

Atau, jika Anda menggunakan OIDC penyedia untuk federasi dan pengguna diautentikasi`AssumeRoleWithWebIdentity`, kebijakan kepercayaan peran Anda mungkin terlihat sebagai berikut.

Example Contoh kebijakan kepercayaan peran untuk identitas sumber (OIDCpenyedia)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

Peran perangkauan dan persyaratan lintas akun

Bayangkan Anda ingin mengizinkan pengguna yang mengasumsikan `CriticalRole` untuk mengambil `CriticalRole_2` di akun lain. Kredensial sesi peran yang diperoleh untuk mengasumsikan `CriticalRole` digunakan untuk merangkai peran untuk peran kedua, `CriticalRole_2`, di akun yang berbeda. Peran sedang diasumsikan di batas akun. Oleh karena itu, izin `sts:SetSourceIdentity` harus diberikan dalam kedua kebijakan izin pada `CriticalRole` dan dalam kebijakan kepercayaan peran di `CriticalRole_2`.

Example Contoh kebijakan izin pada CriticalRole

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

Untuk mengamankan pengaturan sumber identitas di batas akun, kebijakan kepercayaan peran berikut hanya mempercayai peran utama untuk `CriticalRole` guna mengatur identitas sumber.

Example Contoh kebijakan kepercayaan peran pada CriticalRole _2

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

Pengguna membuat panggilan berikut menggunakan kredensi sesi peran yang diperoleh dari asumsi. CriticalRole Identitas sumber ditetapkan selama asumsi CriticalRole, sehingga tidak perlu diatur lagi secara eksplisit. Jika pengguna mencoba untuk mengatur identitas sumber yang berbeda dari nilai yang ditetapkan ketika diasumsikan `CriticalRole`, permintaan peran asumsi akan ditolak.

Example AssumeRole CLIPermintaan contoh

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

Ketika penanggung jawab panggilan mengasumsikan peran, identitas sumber dalam permintaan tetap dari sesi peran diasumsikan pertama. Oleh karena itu, kedua kunci `aws:SourceIdentity` dan `sts:SourceIdentity` terdapat dalam konteks permintaan.

Melihat identitas sumber di CloudTrail

Anda dapat menggunakan CloudTrail untuk melihat permintaan yang dibuat untuk mengambil peran atau pengguna federasi. Anda juga dapat melihat peran atau permintaan pengguna untuk mengambil tindakan dalam AWS. File CloudTrail log mencakup informasi tentang identitas sumber yang ditetapkan untuk peran yang diasumsikan atau sesi pengguna federasi. Untuk informasi selengkapnya, silakan lihat Penebangan IAM dan AWS STS APIpanggilan dengan AWS CloudTrail

Misalnya, asumsikan bahwa pengguna membuat AWS STS `AssumeRole` permintaan, dan menetapkan identitas sumber. Anda dapat menemukan `sourceIdentity` informasi di `requestParameters` kunci di CloudTrail log Anda.

Example Contoh requestParameters bagian dalam AWS CloudTrail log

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

Jika pengguna menggunakan sesi peran yang diasumsikan untuk melakukan tindakan, informasi identitas sumber ada di `userIdentity` kunci di CloudTrail log.

Example Contoh userIdentity kunci dalam AWS CloudTrail log

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```

Untuk melihat contoh AWS STS API peristiwa di CloudTrail log, lihatContoh IAM API peristiwa di CloudTrail log. Untuk detail selengkapnya tentang informasi yang terkandung dalam file CloudTrail log, lihat Referensi CloudTrail Acara di Panduan AWS CloudTrail Pengguna.

Izin untuk GetFederationToken

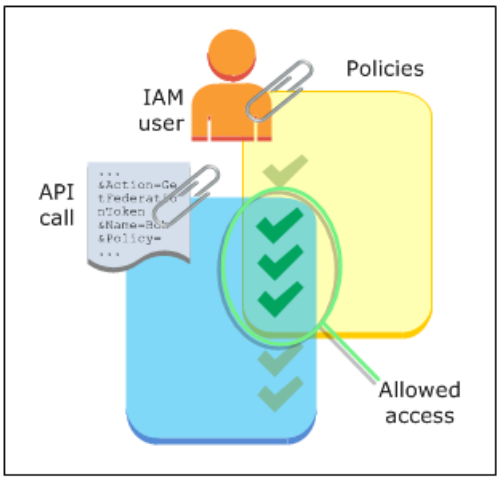
`GetFederationToken`Operasi ini dipanggil oleh IAM pengguna dan mengembalikan kredensi sementara untuk pengguna tersebut. Operasi ini menggabungkan pengguna. Izin yang diberikan pengguna gabungan ditentukan di salah satu dari dua tempat:

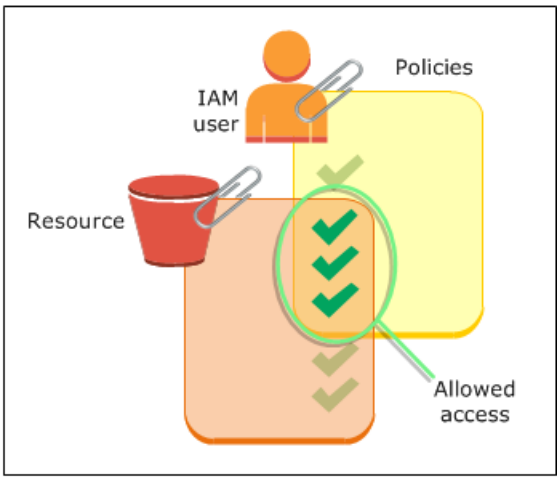
- Kebijakan sesi diteruskan sebagai parameter `GetFederationToken` API panggilan. (Ini paling umum.)
- Kebijakan berbasis sumber daya yang secara jelas menyebutkan pengguna gabungan dalam elemen `Principal` dari kebijakan tersebut. (Ini lebih jarang terjadi.)

Kebijakan sesi adalah kebijakan lanjutan yang Anda jalankan sebagai parameter saat Anda secara terprogram membuat sebuah sesi sementara. Saat Anda membuat sesi pengguna gabungan dan melalui kebijakan sesi, izin sesi yang dihasilkan adalah pertemuan dari kebijakan berbasis pengguna dan kebijakan sesi. Anda tidak dapat menggunakan kebijakan sesi untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari pengguna yang sedang digabung.

Dalam kebanyakan kasus jika Anda tidak meneruskan kebijakan dengan `GetFederationToken` API panggilan, kredensil keamanan sementara yang dihasilkan tidak memiliki izin. Namun, kebijakan berbasis sumber daya dapat memberikan izin tambahan untuk sesi tersebut. Anda dapat mengakses sumber daya dengan sebuah kebijakan berbasis sumber daya yang menentukan sesi Anda sebagai prinsipal yang diizinkan.

Gambar berikut ini menunjukkan gambaran visual tentang cara kebijakan berinteraksi untuk menentukan izin bagi kredensial keamanan sementara yang dikembalikan oleh panggilan ke `GetFederationToken`.





Contoh: Menetapkan izin menggunakan GetFederationToken

Anda dapat menggunakan `GetFederationToken` API tindakan dengan berbagai jenis kebijakan. Berikut adalah beberapa contoh.

Kebijakan yang terlampir pada pengguna IAM

Dalam contoh ini, Anda memiliki aplikasi klien berbasis browser yang mengandalkan dua layanan web backend. Satu layanan backend adalah server autentikasi Anda sendiri yang menggunakan identitas sistem Anda sendiri untuk mengautentikasi aplikasi klien. Layanan backend lainnya adalah layanan AWS yang menyediakan beberapa fungsionalitas aplikasi klien. Aplikasi klien diautentikasi oleh server Anda, dan server Anda membuat atau mengambil kebijakan izin yang sesuai. Server Anda kemudian memanggil `GetFederationToken` API untuk mendapatkan kredensil keamanan sementara, dan mengembalikan kredensil tersebut ke aplikasi klien. Aplikasi klien kemudian dapat membuat permintaan langsung ke AWS layanan dengan kredenal keamanan sementara. Arsitektur ini memungkinkan aplikasi klien untuk membuat AWS permintaan tanpa menyematkan AWS kredensi jangka panjang.

Server otentikasi Anda memanggil `GetFederationToken` API dengan kredensi keamanan jangka panjang dari pengguna yang IAM bernama. `token-app` Tetapi kredensi IAM pengguna jangka panjang tetap ada di server Anda dan tidak pernah didistribusikan ke klien. Contoh kebijakan berikut dilampirkan ke `token-app` IAM pengguna dan menentukan kumpulan izin terluas yang diperlukan oleh pengguna federasi (klien) Anda. Perhatikan bahwa izin `sts:GetFederationToken` diperlukan bagi layanan autentikasi Anda untuk memperoleh kredensial keamanan sementara bagi pengguna yang digabung.

> **Note**
>
> AWS menyediakan contoh aplikasi Java untuk melayani tujuan ini, yang dapat Anda unduh di sini: Mesin Penjual Token untuk Pendaftaran Identitas - Contoh Aplikasi Web Java.

Example Contoh kebijakan yang dilampirkan ke IAM pengguna **`token-app`** yang memanggil **`GetFederationToken`**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
    }
  ]
}
```

Kebijakan sebelumnya memberikan beberapa izin kepada pengguna. IAM Namun, kebijakan ini saja tidak memberikan izin apa pun kepada pengguna gabungan. Jika IAM pengguna ini memanggil `GetFederationToken` dan tidak meneruskan kebijakan sebagai parameter API panggilan, pengguna federasi yang dihasilkan tidak memiliki izin yang efektif.

Kebijakan sesi diberikan sebagai parameter

Cara paling umum untuk memastikan bahwa pengguna federasi diberi izin yang sesuai adalah dengan meneruskan kebijakan sesi dalam `GetFederationToken` API panggilan. Memperluas contoh sebelumnya, bayangkan yang `GetFederationToken` dipanggil dengan kredensil pengguna. IAM `token-app` Kemudian bayangkan bahwa kebijakan sesi berikut diteruskan sebagai parameter API panggilan. Pengguna federasi yang dihasilkan memiliki izin untuk mencantumkan konten bucket Amazon S3 yang diberi nama. `productionapp` Pengguna tidak dapat melakukan Amazon S3 `GetObject``PutObject`, dan `DeleteObject` tindakan pada item di bucket. `productionapp`

Pengguna federasi diberi izin ini karena izin adalah persimpangan kebijakan IAM pengguna dan kebijakan sesi yang Anda lewati.

Pengguna federasi tidak dapat melakukan tindakan di AmazonSNS, Amazon, Amazon DynamoDBSQS, atau di bucket S3 apa pun kecuali. `productionapp` Tindakan ini ditolak meskipun

izin tersebut diberikan kepada IAM pengguna yang terkait dengan `GetFederationToken` panggilan.

Example Contoh kebijakan sesi diteruskan sebagai parameter **`GetFederationToken`** API panggilan

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": ["arn:aws:s3:::productionapp/*"]
    }
  ]
}
```

Kebijakan berbasis sumber daya

Beberapa AWS sumber daya mendukung kebijakan berbasis sumber daya, dan kebijakan ini menyediakan mekanisme lain untuk memberikan izin langsung kepada pengguna federasi. Hanya beberapa AWS layanan yang mendukung kebijakan berbasis sumber daya. Misalnya, Amazon S3 memiliki bucket, Amazon memiliki topik, dan Amazon SNS SQS memiliki antrian yang dapat Anda lampirkan kebijakannya. Untuk daftar semua layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM dan tinjau kolom “Kebijakan berbasis Sumber Daya” pada tabel. Anda dapat menggunakan kebijakan berbasis sumber daya untuk menetapkan izin secara langsung kepada pengguna gabungan. Lakukan ini dengan menentukan Amazon Resource Name (ARN) pengguna federasi dalam `Principal` elemen kebijakan berbasis sumber daya. Contoh berikut menggambarkan hal ini dan memperluas contoh sebelumnya, menggunakan bucket S3 bernama `productionapp`.

Kebijakan berbasis sumber daya berikut ini terlampir di bucket. Kebijakan bucket ini mengijinkan pengguna gabungan bernama Carol untuk mengakses bucket. Ketika kebijakan contoh yang dijelaskan sebelumnya dilampirkan ke `token-app` IAM pengguna, pengguna federasi bernama Carol memiliki izin untuk melakukan`s3:GetObject`,`s3:PutObject`, dan `s3:DeleteObject` tindakan pada bucket bernama. `productionapp` Ini benar bahkan ketika tidak ada kebijakan sesi yang diteruskan sebagai parameter `GetFederationToken` API panggilan. Itu karena dalam kasus ini pengguna gabungan bernama Carol telah diberikan izin secara jelas oleh kebijakan berbasis sumber daya berikut.

Ingat, pengguna federasi diberikan izin hanya jika izin tersebut secara eksplisit diberikan kepada pengguna dan IAM pengguna federasi. Mereka juga dapat diberikan (dalam akun) oleh kebijakan berbasis sumber daya yang secara eksplisit menyebutkan nama pengguna federasi dalam `Principal` elemen kebijakan, seperti pada contoh berikut.

Example Contoh kebijakan bucket yang mengijinkan akses ke pengguna gabungan

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "arn:aws:sts::account-id:federated-user/Carol"},
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::productionapp/*"]
  }
}
```

Untuk informasi selengkapnya tentang bagaimana kebijakan dievaluasi, lihat Logika evaluasi kebijakan.

Izin untuk GetSessionToken

Kesempatan utama untuk memanggil `GetSessionToken` API operasi atau `get-session-token` CLI perintah adalah ketika pengguna harus diautentikasi dengan otentikasi multi-faktor (). MFA Dimungkinkan untuk menulis kebijakan yang memungkinkan tindakan tertentu hanya ketika tindakan tersebut diminta oleh pengguna yang telah diautentikasiMFA. Agar berhasil melewati pemeriksaan MFA otorisasi, pengguna harus terlebih dahulu memanggil `GetSessionToken` dan menyertakan

opsional `SerialNumber` dan `TokenCode` parameter. Jika pengguna berhasil diautentikasi dengan MFA perangkat, kredensional yang dikembalikan oleh `GetSessionToken` API operasi menyertakan konteksnya. MFA Konteks ini menunjukkan bahwa pengguna diautentikasi dengan MFA dan diotorisasi untuk API operasi yang memerlukan MFA otentikasi.

Izin diperlukan untuk GetSessionToken

Tidak ada izin yang diperlukan bagi pengguna untuk mendapatkan token sesi. Tujuan dari `GetSessionToken` operasi ini adalah untuk mengotentikasi pengguna menggunakanMFA. Anda tidak dapat menggunakan kebijakan untuk mengontrol operasi autentikasi.

Untuk memberikan izin untuk melakukan sebagian besar AWS operasi, Anda menambahkan tindakan dengan nama yang sama ke kebijakan. Misalnya, untuk membuat pengguna, Anda harus menggunakan `CreateUser` API operasi, `create-user` CLI perintah, atau AWS Management Console. Untuk melakukan operasi ini, Anda harus memiliki kebijakan yang mengijinkan Anda untuk mengakses tindakan `CreateUser`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateUser",
            "Resource": "*"
        }
    ]
}
```

Anda dapat memasukkan tindakan `GetSessionToken` dalam kebijakan Anda, tetapi tidak itu berpengaruh pada kemampuan pengguna untuk melakukan operasi `GetSessionToken`.

Izin diberikan oleh GetSessionToken

Jika `GetSessionToken` dipanggil dengan kredensi IAM pengguna, kredenal keamanan sementara memiliki izin yang sama dengan pengguna. IAM Demikian pula, jika `GetSessionToken` dipanggil dengan Pengguna root akun AWS kredenal, kredenal keamanan sementara memiliki izin pengguna root.

> **Note**
>
> Kami menyarankan agar Anda tidak memanggil `GetSessionToken` dengan kredensial pengguna root. Sebagai gantinya, ikuti praktik terbaik kami dan buat IAM pengguna dengan izin yang mereka butuhkan. Kemudian gunakan IAM pengguna ini untuk interaksi sehari-hari dengan AWS.

Kredensial sementara yang Anda dapatkan ketika Anda memanggil `GetSessionToken` memiliki kemampuan dan batasan sebagai berikut:

- Anda dapat menggunakan kredensional untuk mengakses AWS Management Console dengan meneruskan kredensialnya ke titik akhir masuk tunggal federasi di. `https://signin.aws.amazon.com/federation` Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.
- Anda tidak dapat menggunakan kredensi untuk memanggil IAM atau AWS STS API operasi. Anda dapat menggunakannya untuk memanggil API operasi untuk AWS layanan lain.

Bandingkan API operasi ini dan keterbatasan serta kemampuannya dengan API operasi lain yang menciptakan kredensi keamanan sementara di Bandingkan AWS STS kredensialnya

Untuk informasi selengkapnya tentang penggunaan API akses yang MFA dilindungi`GetSessionToken`, lihatAPIAkses aman dengan MFA.

Menonaktifkan izin untuk kredensial keamanan sementara

Kredensi keamanan sementara berlaku hingga kedaluwarsa. Kredensi ini berlaku untuk durasi yang ditentukan, dari 900 detik (15 menit) hingga maksimum 129.600 detik (36 jam). Durasi sesi default adalah 43.200 detik (12 jam). Anda dapat mencabut kredensil ini, tetapi Anda juga harus mengubah izin untuk IAM pengguna atau peran untuk menghentikan penggunaan kredensil yang disusupi untuk aktivitas akun berbahaya. Izin yang ditetapkan untuk kredensil keamanan sementara dievaluasi setiap kali digunakan untuk membuat permintaan. AWS Setelah Anda menghapus semua izin dari kredensi, AWS permintaan yang menggunakannya gagal.

Mungkin perlu beberapa menit agar pembaruan kebijakan diterapkan. Untuk sesi IAM peran, Anda dapat mencabut kredensil keamanan sementara peran untuk memaksa semua pengguna yang menganggap peran tersebut mengautentikasi ulang dan meminta kredensil baru. Untuk informasi selengkapnya, lihat Mencabut kredenal keamanan sementara peran tersebut.

Anda tidak dapat mengubah izin untuk file Pengguna root akun AWS. Demikian juga, Anda tidak dapat mengubah izin untuk kredensial keamanan sementara yang dibuat dengan memanggil `GetFederationToken` atau `GetSessionToken` saat masuk sebagai pengguna root. Karena alasan ini, kami menyarankan agar Anda tidak memanggil `GetFederationToken` atau `GetSessionToken` sebagai pengguna root.

Untuk prosedur tentang cara mengubah izin untuk IAM pengguna, lihatMengubah izin untuk pengguna IAM.

Untuk prosedur tentang cara mengubah izin untuk IAM peran, lihatMemperbarui izin untuk peran.

> **Important**
>
> Anda tidak dapat mengedit peran IAM yang dibuat dari set izin Pusat IAM Identitas. Anda harus mencabut sesi set izin aktif untuk pengguna di Pusat IAM Identitas. Untuk informasi selengkapnya, lihat Mencabut sesi IAM peran aktif yang dibuat oleh set izin di Panduan Pengguna Pusat IAM Identitas.

Topik

- Tolak akses ke semua sesi IAM peran yang terkait dengan peran
- Tolak akses ke sesi IAM peran tertentu
- Tolak akses ke sesi kredensi keamanan sementara dengan kunci konteks kondisi
- Tolak akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya

Tolak akses ke semua sesi IAM peran yang terkait dengan peran

Prosedur ini menolak izin untuk semua sesi IAM peran yang terkait dengan peran. Gunakan pendekatan ini ketika Anda khawatir tentang akses yang mencurigakan dengan:

- Prinsipal dari akun lain menggunakan akses lintas akun
- Identitas pengguna eksternal dengan izin untuk mengakses AWS sumber daya di akun Anda
- Pengguna yang telah diautentikasi dalam aplikasi seluler atau web dengan penyedia OIDC

Untuk mengubah atau menghapus izin yang ditetapkan ke kredensil keamanan sementara yang diperoleh dengan memanggil`AssumeRole`,, atau,`AssumeRoleWithSAML`, atau

`AssumeRoleWithWebIdentity`GetFederationToken, atau`GetSessionToken`, Anda dapat mengedit atau menghapus kebijakan berbasis identitas yang menentukan izin untuk peran tersebut.

> **Important**
>
> Jika ada kebijakan berbasis sumber daya yang memungkinkan akses utama, Anda juga harus menambahkan penolakan eksplisit untuk sumber daya tersebut. Lihat Tolak akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya untuk detail.

Untuk menolak akses ke semua sesi IAM peran yang terkait dengan peran

1. Masuk ke AWS Management Console dan buka IAM konsol.
2. Di panel navigasi, pilih Peran..
3. Pilih nama peran yang akan diedit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar.
4. Pilih tab Izin.
5. Pilih kebijakan yang relevan untuk diedit. Sebelum Anda mengedit kebijakan terkelola pelanggan, tinjau tab Entitas yang dilampirkan untuk menghindari gangguan akses ke identitas lain yang mungkin memiliki kebijakan yang sama.
6. Pilih JSONtab dan perbarui kebijakan untuk menolak semua sumber daya dan tindakan.

> **Note**
>
> Izin ini sama dengan yang ada dalam kebijakan AWS AWSDenyAllterkelola. Anda dapat melampirkan kebijakan AWS terkelola ini ke IAM pengguna atau peran mana pun yang ingin Anda tolak semua aksesnya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAll",
            "Effect": "Deny",
            "Action": [
                "*"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

7. Di halaman Tinjauan, tinjau Ringkasan kebijakan lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Saat memperbarui kebijakan, perubahan akan memengaruhi izin semua kredensil keamanan sementara yang terkait dengan peran tersebut, termasuk kredensil yang dikeluarkan sebelum Anda mengubah kebijakan izin peran.

Setelah memperbarui kebijakan, Anda dapat mencabut kredensil keamanan sementara peran tersebut untuk segera mencabut semua izin atas kredenal yang dikeluarkan peran tersebut.

Tolak akses ke sesi IAM peran tertentu

Bila Anda memperbarui IAM peran dengan kebijakan tolak semua atau menghapus peran sepenuhnya, semua pengguna yang memiliki akses ke peran akan terganggu. Anda dapat menolak akses tanpa memengaruhi izin semua sesi lain yang terkait dengan peran tersebut.

Izin `Principal` dapat ditolak menggunakan kunci konteks kondisi atau kebijakan berbasis sumber daya.

> **Tip**
>
> Anda dapat menemukan pengguna ARNs federasi menggunakan AWS CloudTrail log. Untuk informasi selengkapnya, lihat Cara Mudah Mengidentifikasi Pengguna Federasi Anda dengan Menggunakan AWS CloudTrail.

Tolak akses ke sesi kredensi keamanan sementara dengan kunci konteks kondisi

Anda dapat menggunakan kunci konteks kondisi dalam kebijakan berbasis identitas dalam situasi di mana Anda ingin menolak akses ke sesi kredensi keamanan sementara tertentu tanpa memengaruhi izin IAM pengguna atau peran yang membuat kredensil. Untuk IAM peran, setelah memperbarui kebijakan, Anda juga dapat mencabut sesi kredensil keamanan sementara peran tersebut untuk segera mencabut semua kredensi yang dikeluarkan.

Untuk informasi selengkapnya tentang kunci konteks kondisi, lihatAWS kunci konteks kondisi global.

aws: PrincipalArn

Anda dapat menggunakan kunci konteks kondisi aws: PrincipalArn dalam kebijakan berbasis identitas untuk menolak akses ke prinsipal tertentu dengan Amazon Resource Name (). ARN Anda melakukannya dengan menentukan sesi IAM pengguna, peran, atau pengguna AWS STS gabungan yang terkait dengan kredensil keamanan sementara dalam elemen Kondisi kebijakan. ARN

Untuk menolak akses ke kepala sekolah tertentu oleh mereka ARN

1. Di panel navigasi IAM konsol, pilih Pengguna atau Peran.
2. Pilih nama IAM pengguna atau peran yang akan diedit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar.
3. Pilih tab Izin.
4. Pilih kebijakan yang relevan untuk diedit. Sebelum Anda mengedit kebijakan terkelola pelanggan, tinjau tab Entitas yang dilampirkan untuk menghindari gangguan akses ke identitas lain yang mungkin memiliki kebijakan yang sama.
5. Pilih JSONtab dan tambahkan pernyataan penolakan untuk prinsipal ARN seperti yang ditunjukkan pada contoh berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:iam::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
  ]
}
```

6. Di halaman Tinjauan, tinjau Ringkasan kebijakan lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

aws: SourceIdentity

Anda dapat menggunakan kunci konteks kondisi aws: SourceIdentity dalam kebijakan berbasis identitas untuk menolak akses ke identitas sumber tertentu yang terkait dengan sesi peranIAM. Ini berlaku selama sesi peran dikeluarkan dengan menyetel parameter `SourceIdentity` permintaan saat prinsipal mengambil peran menggunakan AWS STS `assume-role` CLI perintah*, atau AWS STS `AssumeRole` API operasi* apa pun. Anda melakukannya dengan menentukan identitas sumber yang terkait dengan kredensi keamanan sementara dalam `Condition` elemen kebijakan.

Tidak seperti kunci konteks sts:RoleSessionName, setelah identitas sumber ditetapkan, nilainya tidak dapat diubah. `aws:SourceIdentity`Kuncinya ada dalam konteks permintaan untuk semua tindakan yang diambil oleh peran. Identitas sumber tetap ada di sesi peran berikutnya saat Anda menggunakan kredensil sesi untuk mengambil peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

Kebijakan berikut menunjukkan contoh bagaimana Anda dapat menolak akses ke sesi kredensi keamanan sementara menggunakan kunci `aws:SourceIdentity` konteks kondisi. Jika Anda menentukan identitas sumber yang terkait dengan sesi peran, itu akan menolak sesi peran dengan identitas sumber bernama tanpa memengaruhi izin peran yang membuat kredensil. Untuk contoh ini, identitas sumber yang ditetapkan oleh kepala sekolah saat sesi peran dikeluarkan adalah`nikki_wolf@example.com`. Setiap permintaan yang dibuat oleh sesi peran dengan identitas sumber `nikki_wolf@example.com` akan ditolak karena identitas sumber disertakan dalam kondisi kebijakan dan Efek kebijakan disetel ke`Deny`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": [
            "nikki_wolf@example.com",
            "<source identity value>"
          ]
```

```
        }
      }
    }
  ]
}
```

aws:userid

Anda dapat menggunakan kunci konteks kondisi aws:userid dalam kebijakan berbasis identitas untuk menolak akses ke semua atau sesi kredensi keamanan sementara tertentu yang terkait dengan pengguna atau peran. IAM Anda melakukannya dengan menentukan pengenal unik (ID) IAM pengguna, peran, atau sesi pengguna AWS STS gabungan yang terkait dengan kredensil keamanan sementara dalam elemen kebijakan. `Condition`

Kebijakan berikut menunjukkan contoh bagaimana Anda dapat menolak akses ke sesi kredensi keamanan sementara menggunakan kunci `aws:userid` konteks kondisi.

- `AIDAXUSER1`mewakili ID unik untuk IAM pengguna. Menentukan ID unik IAM pengguna sebagai nilai untuk kunci konteks `aws:userid` akan menolak akses ke IAM pengguna. Ini termasuk sesi kredensi keamanan sementara yang dibuat dengan menelepon. `GetSessionToken` API
- `AROAXROLE1:*`mewakili ID unik untuk semua sesi yang terkait dengan IAM peran. Menentukan ID unik IAM peran dan karakter wildcard (*) di bagian caller-specified-role-session -name sebagai nilai untuk kunci konteks `aws:userid` akan menolak semua sesi yang terkait dengan peran tersebut.
- `AROAXROLE2:<caller-specified-role-session-name>`mewakili ID unik untuk sesi peran yang diasumsikan. Di bagian caller-specified-role-session -name dari ID unik peran yang diasumsikan, Anda dapat menentukan nama sesi peran atau karakter wildcard jika operator kondisi digunakan. StringLike Jika Anda menentukan nama sesi peran, itu akan menolak sesi peran bernama tanpa memengaruhi izin peran yang membuat kredensil. Jika Anda menentukan karakter wildcard untuk nama sesi peran, itu akan menolak semua sesi yang terkait dengan peran tersebut.

> **Note**
>
> Nama sesi peran yang ditentukan pemanggil, yang merupakan bagian dari pengenal unik untuk sesi peran yang diasumsikan, dapat berubah selama rantai peran. Role chaining terjadi ketika satu peran mengambil peran lain. Nama sesi peran diatur menggunakan parameter `RoleSessionName` permintaan ketika prinsipal mengasumsikan peran menggunakan AWS STS `AssumeRole` API operasi.

- `account-id:<federated-user-caller-specified-name>`mewakili ID unik untuk sesi pengguna AWS STS federasi. Seorang IAM pengguna membuat sesi ini dengan memanggil file `GetFederationToken`API. Menentukan ID unik untuk sesi pengguna AWS STS federasi menyangkal sesi pengguna federasi bernama tanpa memengaruhi izin pengguna yang membuat kredensialnyaIAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1:*",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "account-id:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
}
```

Untuk contoh spesifik dari nilai kunci utama, lihatNilai-nilai kunci utama. Untuk informasi tentang pengidentifikasi IAM unik dan cara mendapatkannya, lihatPengidentifikasi unik.

Tolak akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya

Untuk membatasi akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya, Anda dapat menggunakan kunci aws: PrincipalArn konteks kondisi atau dalam elemen. aws: SourceIdentity `Condition` Kebijakan berbasis sumber daya adalah kebijakan izin yang dilampirkan pada sumber daya dan kontrol siapa yang dapat mengakses sumber daya dan tindakan apa yang dapat mereka lakukan terhadapnya.

Saat Anda menggunakan kunci `aws:PrincipalARN` konteks, tentukan sesi IAM pengguna, peran, atau pengguna AWS STS gabungan yang terkait dengan kredensil keamanan sementara dalam

elemen Kondisi kebijakan. ARN Contoh kebijakan berikut menunjukkan cara menggunakan kunci `aws:PrincipalARN` konteks dalam kebijakan berbasis sumber daya:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": [
      "*"
    ],
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "ArnEquals": {
        "aws:PrincipalArn": [
          "arn:aws:iam::222222222222:role/ROLENAME",
          "arn:aws:iam::222222222222:user/USERNAME",
          "arn:aws:sts::222222222222:federated-user/USERNAME"
        ]
      }
    }
  }
}
```

Bila Anda menggunakan kunci `aws:SourceIdentity` konteks, tentukan nilai identitas sumber yang terkait dengan kredensil keamanan sementara peran dalam `Condition` elemen kebijakan. Ini berlaku selama sesi peran dikeluarkan dengan menyetel parameter `SourceIdentity` permintaan saat prinsipal mengambil peran menggunakan AWS STS `assume-role` CLI perintah*, atau AWS STS `AssumeRole` API operasi* apa pun. Contoh berikut menunjukkan cara menggunakan kunci `aws:SourceIdentity` konteks dalam kebijakan berbasis sumber daya:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": [
      "*"
    ],
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "StringLike": {
```

```
        "aws:SourceIdentity": [
          "nikki_wolf@example.com",
          "<source identity value>"
        ]
      }
    }
  }
}
```

Jika Anda hanya memperbarui kebijakan berbasis identitas untuk prinsipal, mereka masih dapat melakukan tindakan yang diizinkan dalam kebijakan berbasis sumber daya, kecuali jika tindakan tersebut secara eksplisit ditolak dalam kebijakan berbasis identitas.

Untuk menolak akses ke prinsipal tertentu dalam kebijakan berbasis sumber daya

1. Lihat AWS layanan yang bekerja dengan IAM untuk melihat apakah layanan mendukung kebijakan berbasis sumber daya.
2. Masuk ke AWS Management Console dan buka konsol untuk layanan ini. Setiap layanan memiliki lokasi yang berbeda di konsol untuk melampirkan kebijakan.
3. Edit kebijakan berbasis sumber daya. Tambahkan pernyataan kebijakan penolakan untuk menentukan informasi identifikasi kredensi:

   a. Dalam `Principal` elemen, masukkan wildcard (*). Kepala sekolah akan dibatasi dalam `Condition` elemen.

   b. Dalam `Effect` elemen, masukkan “Deny.”

   c. Masuk`Action`, masukkan namespace layanan dan nama tindakan yang akan ditolak. Untuk menolak semua tindakan, gunakan karakter wildcard (*). Sebagai contoh: `"s3:*"`.

   d. Dalam `Resource` elemen, masukkan ARN sumber daya target. Sebagai contoh: `"arn:aws:s3:::amzn-s3-demo-bucket"`.

   e. Dalam `Condition` elemen, tentukan kunci `aws:PrincipalARN` atau `aws:SourceIdentity` konteks.

      Jika Anda menggunakan kunci `aws:PrincipalARN` konteks, masukkan ARN prinsipal untuk menolak akses.

      Jika Anda menggunakan kunci `aws:SourceIdentity` konteks, masukkan nilai identitas sumber yang ditetapkan dalam sesi peran untuk menolak akses.

4. Simpan pekerjaan Anda.

Memberikan izin untuk membuat kredensial keamanan sementara

Secara default, pengguna IAM tidak memiliki izin untuk membuat kredensial keamanan sementara untuk pengguna gabungan dan peran. Anda harus menggunakan kebijakan untuk memberikan izin ini kepada pengguna Anda. Meskipun Anda dapat memberikan izin secara langsung kepada pengguna, kami sangat menyarankan agar Anda memberikan izin kepada grup. Ini menjadikan pengelolaan izin jauh lebih mudah. Saat seseorang tidak lagi perlu melakukan tugas yang terkait dengan izin, Anda cukup menghapusnya dari grup. Jika orang lain perlu melakukan tugas tersebut, tambahkan mereka ke grup untuk memberikan izin.

Untuk memberikan izin grup IAM untuk membuat kredensial keamanan sementara bagi pengguna gabungan atau peran, Anda melampirkan kebijakan yang memberikan salah satu atau kedua hak istimewa berikut:

- Untuk pengguna federasi untuk mengakses peran IAM, berikan akses ke. AWS STS `AssumeRole`
- Untuk pengguna federasi yang tidak memerlukan peran, berikan akses ke AWS STS `GetFederationToken`.

Untuk informasi selengkapnya tentang perbedaan antara operasi API `AssumeRole` dan `GetFederationToken`, lihat Minta kredensi keamanan sementara.

Pengguna IAM juga dapat memanggil `GetSessionToken` untuk membuat kredensial keamanan sementara. Tidak ada izin yang diperlukan bagi pengguna untuk memanggil `GetSessionToken`. Tujuan dari operasi ini adalah mengautentikasi pengguna menggunakan MFA. Anda tidak dapat menggunakan kebijakan untuk mengontrol autentikasi. Ini artinya Anda tidak dapat mencegah pengguna IAM melakukan panggilan `GetSessionToken` untuk membuat kredensial sementara.

Example Contoh kebijakan yang memberikan izin untuk mengambil peran

Contoh kebijakan berikut memberikan izin `AssumeRole` untuk memanggil `UpdateApp` peran dalam Akun AWS `123123123123`. Saat `AssumeRole` digunakan, pengguna (atau aplikasi) yang membuat kredensial keamanan atas nama pengguna gabungan tidak dapat mengurangi izin apapun yang belum ditentukan dalam kebijakan izin peran.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
```

```
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example Contoh kebijakan yang memberikan izin untuk membuat kredensial keamanan sementara bagi pengguna gabungan.

Contoh kebijakan berikut memberikan izin untuk mengakses `GetFederationToken`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

> **Important**
>
> Saat Anda memberikan pengguna IAM izin untuk membuat kredensial keamanan sementara bagi pengguna gabungan dengan `GetFederationToken`, ketahuilah bahwa ini mengizinkan pengguna tersebut untuk mengeluarkan izin milik mereka. Untuk informasi selengkapnya tentang mendelegasikan izin di seluruh pengguna IAM dan Akun AWS, lihat. Contoh kebijakan untuk mendelegasikan akses Untuk informasi selengkapnya tentang mengatur izin pada kredensial keamanan sementara, lihat Izin untuk kredensial keamanan sementara.

Example Contoh kebijakan yang memberikan izin terbatas kepada pengguna untuk membuat kredensial keamanan sementara bagi pengguna gabungan.

Saat Anda mengizinkan pengguna IAM memanggil `GetFederationToken`, ini adalah penerapan terbaik untuk membatasi izin yang dapat dikeluarkan oleh pengguna IAM. Misalnya, kebijakan berikut menunjukkan cara membiarkan pengguna IAM membuat kredensial keamanan sementara hanya untuk pengguna gabungan yang namanya dimulai dengan Manajer.

```
{
  "Version": "2012-10-17",
  "Statement": [{
```

```
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Memberikan izin untuk menggunakan sesi konsol sadar identitas

Sesi konsol yang sadar identitas memungkinkan ID AWS IAM Identity Center pengguna dan sesi disertakan dalam sesi AWS konsol pengguna saat mereka masuk. Misalnya, Amazon Q Developer Pro menggunakan sesi konsol sadar identitas untuk mempersonalisasi pengalaman layanan. Untuk informasi selengkapnya tentang sesi konsol sadar identitas, lihat Mengaktifkan sesi konsol sadar identitas di Panduan Pengguna.AWS IAM Identity Center Untuk informasi tentang penyiapan Pengembang Amazon Q, lihat Menyiapkan Pengembang Amazon Q di Panduan Pengguna Pengembang Amazon Q.

Agar sesi konsol sadar identitas tersedia bagi pengguna, Anda harus menggunakan kebijakan berbasis identitas untuk memberikan `sts:SetContext` izin kepada prinsipal IAM untuk sumber daya yang mewakili sesi konsol mereka sendiri.

> **Important**
>
> Secara default, pengguna tidak memiliki izin untuk menyetel konteks untuk sesi konsol sadar identitas mereka. Untuk mengizinkan hal ini, Anda harus memberikan `sts:SetContext` izin kepada kepala IAM dalam kebijakan berbasis identitas seperti yang ditunjukkan pada contoh kebijakan di bawah ini.

Contoh kebijakan berbasis identitas berikut memberikan `sts:SetContext` izin kepada prinsipal IAM, memungkinkan prinsipal untuk mengatur konteks sesi konsol sadar identitas untuk sesi konsol mereka sendiri. AWS Sumber daya kebijakan,`arn:aws:sts::`*`account-id`*`:self`, mewakili AWS sesi pemanggil. Segmen `account-id` ARN dapat diganti dengan karakter * wildcard jika kebijakan izin yang sama diterapkan di beberapa akun, seperti saat kebijakan ini diterapkan menggunakan set izin Pusat Identitas IAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": "sts:SetContext",
      "Resource": "arn:aws:sts::account-id:self"
    }
  ]
}
```

Kelola AWS STS dalam sebuah Wilayah AWS

Titik akhir Regional adalah titik masuk dalam wilayah tertentu untuk URL AWS layanan web. AWS merekomendasikan menggunakan Regional AWS Security Token Service (AWS STS) titik akhir alih-alih titik akhir global untuk mengurangi latensi, membangun redundansi, dan meningkatkan validitas token sesi. Meskipun global (warisan) AWS STS titik akhir `https://sts.amazonaws.com` sangat tersedia, dihosting dalam satu AWS Wilayah, AS Timur (Virginia N.), dan seperti titik akhir lainnya, ia tidak menyediakan failover otomatis ke titik akhir di Wilayah lain.

- Kurangi latensi — Dengan membuat AWS STS panggilan ke titik akhir yang secara geografis lebih dekat dengan layanan dan aplikasi Anda, Anda dapat mengakses AWS STS layanan dengan latensi yang lebih rendah dan waktu respons yang lebih baik.
- Membangun redundansi — Anda dapat membatasi efek kegagalan dalam beban kerja ke sejumlah komponen terbatas dengan cakupan penahanan dampak yang dapat diprediksi. Menggunakan regional AWS STS endpoint memungkinkan Anda menyelaraskan cakupan komponen Anda dengan cakupan token sesi Anda. Untuk informasi selengkapnya tentang pilar keandalan ini, lihat Menggunakan isolasi kesalahan untuk melindungi beban kerja Anda di AWS Kerangka Well-Architected.
- Meningkatkan validitas token sesi — Token sesi dari Regional AWS STS titik akhir berlaku di semua Wilayah AWS. Token sesi dari STS titik akhir global hanya berlaku di Wilayah AWS yang diaktifkan secara default. Jika Anda bermaksud mengaktifkan Wilayah baru untuk akun Anda, Anda dapat menggunakan token sesi dari Regional AWS STS titik akhir. Jika Anda memilih untuk menggunakan titik akhir global, Anda harus mengubah kompatibilitas Wilayah AWS STS token sesi untuk titik akhir global. Melakukannya memastikan bahwa token valid di semua Wilayah AWS.

Untuk daftar AWS STS Daerah dan titik akhirnya, lihatAWS STS Wilayah dan titik akhir.

Topik

- Mengaktifkan dan menonaktifkan AWS STS dalam sebuah Wilayah AWS
- Menulis kode untuk digunakan AWS STS Wilayah
- Mengelola token sesi titik akhir global

Mengaktifkan dan menonaktifkan AWS STS dalam sebuah Wilayah AWS

Saat Anda mengaktifkan STS titik akhir untuk Wilayah, AWS STS dapat mengeluarkan kredensi sementara untuk pengguna dan peran di akun Anda yang membuat AWS STS permintaan. Kredensial tersebut kemudian dapat digunakan di setiap Wilayah yang diaktifkan secara default atau diaktifkan secara manual. Untuk Wilayah yang diaktifkan secara default, Anda harus mengaktifkan STS titik akhir Regional di akun tempat kredensi sementara dibuat. Tidak masalah apakah pengguna masuk ke akun yang sama atau akun yang berbeda saat mereka mengajukan permintaan. Untuk Wilayah yang diaktifkan secara manual, Anda harus mengaktifkan Wilayah di kedua akun yang membuat permintaan dan akun tempat kredensi sementara dibuat.

Misalnya, bayangkan pengguna di akun A ingin mengirim `sts:AssumeRole` API permintaan ke AWS STS Titik akhir `https://sts.us-west-2.amazonaws.com` regional. Permintaan adalah untuk kredensial sementara untuk peran bernama `Developer` di akun B. karena permintaan adalah untuk membuat kredensial untuk entitas di akun B, akun B harus mengaktifkan Wilayah `us-west-2`. Pengguna dari akun A (atau akun lainnya) dapat menghubungi `us-west-2` AWS STS endpoint untuk meminta kredensil untuk akun B apakah Wilayah diaktifkan di akun mereka atau tidak.

> **Note**
>
> Wilayah Aktif tersedia bagi semua orang yang menggunakan kredensial sementara dalam akun tersebut. Untuk mengontrol IAM pengguna atau peran mana yang dapat mengakses Wilayah, gunakan kunci `aws:RequestedRegion` kondisi dalam kebijakan izin Anda.

Untuk mengaktifkan atau menonaktifkan AWS STS di Wilayah yang diaktifkan secara default (konsol)

1. Masuk sebagai pengguna root atau pengguna dengan izin untuk melakukan tugas IAM administrasi.
2. Buka IAMkonsol dan di panel navigasi pilih Pengaturan akun.
3. Di bagian Security Token Service (STS) Endpoints, cari Region yang ingin Anda konfigurasikan, lalu pilih Aktif atau Tidak Aktif di kolom STSstatus.
4. Di kotak dialog yang terbuka, pilih Aktifkan atau Nonaktifkan.

Untuk Wilayah yang harus diaktifkan, kami aktifkan AWS STS secara otomatis saat Anda mengaktifkan Region. Setelah Anda mengaktifkan Region, AWS STS selalu aktif untuk Wilayah dan Anda tidak dapat menonaktifkannya. Untuk mempelajari cara mengaktifkan Wilayah yang

dinonaktifkan secara default, lihat Menentukan yang Wilayah AWS Akun Anda dapat digunakan di AWS Account Management Panduan Referensi.

Menulis kode untuk digunakan AWS STS Wilayah

Setelah Anda mengaktifkan Region, Anda dapat mengarahkan AWS STS APIpanggilan ke wilayah tersebut. Cuplikan kode Java berikut menunjukkan cara mengkonfigurasi `AWSSecurityTokenService` objek untuk membuat permintaan ke Eropa (Milan) (eu-south-1) Wilayah.

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
south-1.amazonaws.com", "eu-south-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STS merekomendasikan agar Anda melakukan panggilan ke titik akhir Regional. Untuk mempelajari cara mengaktifkan Region secara manual, lihat Menentukan yang Wilayah AWS Akun Anda dapat digunakan di AWS Account Management Panduan Referensi.

Dalam contoh, baris pertama membuat instance `EndpointConfiguration` objek yang dipanggil`regionEndpointConfig`, melewati titik URL akhir dan Wilayah AWS sebagai parameternya.

Untuk mempelajari cara mengatur AWS STS titik akhir regional menggunakan variabel lingkungan untuk AWS SDKs, lihat AWS STS Titik akhir regional di AWS SDKsdan Panduan Referensi Alat.

Untuk semua kombinasi bahasa dan lingkungan pemrograman lainnya, lihat dokumentasi untuk yang relevan SDK.

Mengelola token sesi titik akhir global

Kebanyakan Wilayah AWS diaktifkan untuk operasi di semua Layanan AWS secara default. Wilayah tersebut secara otomatis diaktifkan untuk digunakan dengan AWS STS. Beberapa Wilayah, seperti Asia Pasifik (Hong Kong), harus diaktifkan secara manual. Untuk mempelajari lebih lanjut tentang mengaktifkan dan menonaktifkan Wilayah AWS, lihat Tentukan yang Wilayah AWS Akun Anda dapat digunakan di AWS Account Management Panduan Referensi. Saat Anda mengaktifkan ini

AWS Wilayah, mereka secara otomatis diaktifkan untuk digunakan dengan AWS STS. Anda tidak dapat mengaktifkan AWS STS titik akhir untuk Wilayah yang dinonaktifkan. Token sesi yang valid di semua Wilayah AWS menyertakan lebih banyak karakter daripada token yang valid di Wilayah yang diaktifkan secara default. Mengubah pengaturan ini mungkin memengaruhi sistem yang sudah ada di mana Anda menyimpan token untuk sementara waktu.

Anda dapat mengubah pengaturan ini menggunakan AWS Management Console, AWS CLI, atau AWS API.

Untuk mengubah kesesuaian Wilayah token sesi untuk titik akhir global (konsole)

1. Masuk sebagai pengguna root atau pengguna dengan izin untuk melakukan tugas IAM administrasi. Untuk mengubah kesesuaian token sesi, Anda harus memiliki kebijakan yang memungkinkan tindakan `iam:SetSecurityTokenServicePreferences`.
2. Buka IAMkonsol. Di panel navigasi, pilih Pengaturan akun.
3. Di bawah Security Token Service (STS) bagian Token Sesi dari STS titik akhir. Titik akhir Global menunjukkan`Valid only in Wilayah AWS enabled by default`. Pilih Ubah.
4. Dalam kotak dialog Ubah kompatibilitas wilayah, pilih Semua Wilayah AWS. Kemudian pilih Simpan perubahan.

> **Note**
>
> Token sesi yang valid di semua Wilayah AWS menyertakan lebih banyak karakter daripada token yang valid di Wilayah yang diaktifkan secara default. Mengubah pengaturan ini mungkin memengaruhi sistem yang sudah ada di mana Anda menyimpan token untuk sementara waktu.

Untuk mengubah kompatibilitas Wilayah token sesi untuk titik akhir global (AWS CLI)

Atur versi token sesi. Token versi 1 hanya berlaku di Wilayah AWS yang tersedia secara default. Token ini tidak bekerja di Wilayah yang diaktifkan secara manual, seperti Asia Pasifik (Hong Kong). Token Versi 2 valid di semua Wilayah. Namun, token versi 2 memuat lebih banyak karakter dan mungkin memengaruhi sistem tempat Anda menyimpan token untuk sementara waktu.

- `aws iam set-security-token-service-preferences`

Untuk mengubah kompatibilitas Wilayah token sesi untuk titik akhir global (AWS API)

Atur versi token sesi. Token versi 1 hanya berlaku di Wilayah AWS yang tersedia secara default. Token ini tidak bekerja di Wilayah yang diaktifkan secara manual, seperti Asia Pasifik (Hong Kong). Token Versi 2 valid di semua Wilayah. Namun, token versi 2 memuat lebih banyak karakter dan mungkin memengaruhi sistem tempat Anda menyimpan token untuk sementara waktu.

- `SetSecurityTokenServicePreferences`

AWS STS Wilayah dan titik akhir

Tabel berikut mencantumkan Wilayah dan titik akhirnya. Ini menunjukkan mana yang diaktifkan secara default dan mana yang dapat Anda aktifkan atau nonaktifkan.

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
--Global--	sts.amazonaws.com	 Ya	 Tidak
AS Timur (Ohio)	sts.us-east-2.amazonaws.com	 Ya	 Ya
AS Timur (Virginia Utara)	sts.us-east-1.amazonaws.com	 Ya	 Tidak
AS Barat (California Utara)	sts.us-west-1.amazonaws.com	 Ya	 Ya
AS Barat (Oregon)	sts.us-west-2.amazonaws.com	 Ya	 Ya

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Afrika (Cape Town)	sts.af-south-1.amazonaws.com	Tidak	Tidak
Asia Pasifik (Hong Kong)	sts.ap-east-1.amazonaws.com	Tidak	Tidak
Asia Pasifik (Hyderabad)	sts.ap-south-2.amazonaws.com	Tidak	Tidak
Asia Pasifik (Jakarta)	sts.ap-southeast-3.amazonaws.com	Tidak	Tidak
Asia Pasifik (Melbourne)	sts.ap-southeast-4.amazonaws.com	Tidak	Tidak
Asia Pasifik (Mumbai)	sts.ap-south-1.amazonaws.com	Ya	Ya
Asia Pasifik (Osaka)	sts.ap-northeast-3.amazonaws.com	Ya	Ya
Asia Pasifik (Seoul)	sts.ap-northeast-2.amazonaws.com	Ya	Ya

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Asia Pasifik (Singapura)	sts.ap-southeast-1.amazonaws.com	Ya	Ya
Asia Pasifik (Sydney)	sts.ap-southeast-2.amazonaws.com	Ya	Ya
Asia Pasifik (Tokyo)	sts.ap-northeast-1.amazonaws.com	Ya	Ya
Kanada (Pusat)	sts.ca-central-1.amazonaws.com	Ya	Ya
Kanada Barat (Calgary)	sts.ca-west-1.amazonaws.com	Ya	Ya
Tiongkok (Beijing)	sts.cn-north-1.amazonaws.com.cn	Ya[2]	Tidak
Tiongkok (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	Ya[2]	Ya
Eropa (Frankfurt)	sts.eu-central-1.amazonaws.com	Ya	Ya

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Eropa (Irlandia)	sts.eu-west-1.amazonaws.com	Ya	Ya
Eropa (London)	sts.eu-west-2.amazonaws.com	Ya	Ya
Eropa (Milan)	sts.eu-south-1.amazonaws.com	Tidak	Tidak
Eropa (Paris)	sts.eu-west-3.amazonaws.com	Ya	Ya
Eropa (Spanyol)	sts.eu-south-2.amazonaws.com	Tidak	Tidak
Eropa (Stockholm)	sts.eu-north-1.amazonaws.com	Ya	Ya
Eropa (Zürich)	sts.eu-central-2.amazonaws.com	Tidak	Tidak
Israel (Tel Aviv)	sts.il-central-1.amazonaws.com	Tidak	Tidak

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Timur Tengah (Bahrain)	sts.me-south-1.amazonaws.com	Tidak	Tidak
Timur Tengah (UAE)	sts.me-central-1.amazonaws.com	Tidak	Tidak
Amerika Selatan (Sao Paulo)	sts.sa-east-1.amazonaws.com	Ya	Ya

[1]Anda harus mengaktifkan Wilayah untuk menggunakannya. Ini secara otomatis mengaktifkan AWS STS. Anda tidak dapat mengaktifkan atau menonaktifkan secara manual AWS STS di daerah-daerah tersebut.

[2]Untuk menggunakan AWS di China, Anda memerlukan akun dan kredensi khusus untuk AWS di China.

AWS CloudTrail dan titik akhir Regional

Panggilan ke titik akhir regional dan global dicatat di `tlsDetails` lapangan di AWS CloudTrail. Panggilan ke titik akhir regional, seperti`us-east-2.amazonaws.com`, masuk CloudTrail ke wilayah yang sesuai. Memanggil ke titik akhir global, `sts.amazonaws.com`, dicatat sebagai panggilan ke layanan global. Acara untuk global AWS STS titik akhir dicatat ke us-east-1.

> **Note**
>
> `tlsDetails`hanya dapat dilihat untuk layanan yang mendukung bidang ini. Lihat Layanan yang mendukung TLS detail CloudTrail di AWS CloudTrail Panduan Pengguna

Untuk informasi selengkapnya, lihat Penebangan IAM dan AWS STS APIpanggilan dengan AWS CloudTrail.

Aktifkan akses broker identitas khusus ke AWS konsol

Anda dapat menulis dan menjalankan kode untuk membuat kode URL yang memungkinkan pengguna yang masuk ke jaringan organisasi Anda mengakses dengan aman. AWS Management Console URLTermasuk token masuk yang Anda dapatkan AWS dan yang mengautentikasi pengguna. AWS Sesi konsol yang dihasilkan mungkin termasuk yang berbeda `AccessKeyId` karena federasi. Untuk melacak penggunaan kunci akses untuk login federasi melalui CloudTrail acara terkait, lihat Penebangan IAM dan AWS STS APIpanggilan dengan AWS CloudTrail dan acara AWS Management Console login.

> **Note**
>
> Jika organisasi Anda menggunakan penyedia identitas (iDP) yang kompatibel dengannyaSAML, Anda dapat mengatur akses ke konsol tanpa menulis kode. Ini berfungsi dengan penyedia seperti Layanan Federasi Direktori Aktif Microsoft atau Shibboleth sumber terbuka. Untuk detailnya, lihat Mengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console.

Untuk memungkinkan pengguna organisasi Anda mengakses AWS Management Console, Anda dapat membuat pialang identitas khusus yang melakukan langkah-langkah berikut:

1. Pastikan bahwa pengguna diautentikasi oleh sistem identitas lokal Anda.
2. Hubungi AWS Security Token Service (AWS STS) AssumeRole(disarankan) atau GetFederationTokenAPIoperasi untuk mendapatkan kredenal keamanan sementara bagi pengguna. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk menjalankan peran, lihat Metode untuk mengambil peran. Untuk mempelajari cara mengirimkan tag sesi opsional saat Anda mendapatkan kredensial keamanan Anda, lihat Lulus tag sesi di AWS STS.
   - Jika Anda menggunakan salah satu `AssumeRole*` API operasi untuk mendapatkan kredenal keamanan sementara untuk suatu peran, Anda dapat menyertakan `DurationSeconds` parameter dalam panggilan Anda. Parameter ini menentukan durasi sesi peran Anda, dari 900 detik (15 menit) hingga setelan durasi sesi maksimum untuk peran tersebut. Ketika Anda

menggunakan `DurationSeconds` dalam `AssumeRole*` operasi, Anda harus menyebutnya sebagai IAM pengguna dengan kredensi jangka panjang. Jika tidak, panggilan ke titik akhir federasi pada langkah 3 gagal. Untuk mempelajari cara melihat atau mengubah nilai maksimum untuk peran, lihat Memperbarui durasi sesi maksimum untuk peran.

- Jika Anda menggunakan `GetFederationToken` API operasi untuk mendapatkan kredensialnya, Anda dapat menyertakan `DurationSeconds` parameter dalam panggilan Anda. Parameter ini menentukan durasi sesi peran Anda. Nilai dapat berkisar dari 900 detik (15 menit) hingga 129.600 detik (36 jam). Anda dapat melakukan API panggilan ini hanya dengan menggunakan kredensi AWS keamanan jangka panjang dari pengguna. IAM Anda juga dapat melakukan panggilan ini menggunakan Pengguna root akun AWS kredensional, tetapi kami tidak merekomendasikannya. Jika Anda membuat panggilan ini sebagai pengguna akar, sesi default berlangsung selama satu jam. Atau Anda dapat menentukan sesi dari 900 detik (15 menit) hingga 3.600 detik (satu jam).

3. Hubungi titik akhir AWS federasi dan berikan kredensi keamanan sementara untuk meminta token masuk.
4. Buat a URL untuk konsol yang menyertakan token:
   - Jika Anda menggunakan salah satu `AssumeRole*` API operasi di AndaURL, Anda dapat memasukkan `SessionDuration` HTTP parameter. Parameter ini menentukan durasi sesi konsol, dari 900 detik (15 menit) hingga 43200 detik (12 jam).
   - Jika Anda menggunakan `GetFederationToken` API operasi di AndaURL, Anda dapat memasukkan `DurationSeconds` parameter. Parameter ini menentukan durasi sesi konsol federasi. Nilai dapat berkisar dari 900 detik (15 menit) hingga 129.600 detik (36 jam).

   > **Note**
   >
   > - Jangan gunakan `SessionDuration` HTTP parameter jika Anda mendapatkan kredensi sementara dengan. `GetFederationToken` Melakukan hal itu akan menyebabkan kegagalan operasi.
   > - Menggunakan kredensi untuk satu peran untuk mengambil peran yang berbeda disebut rantai peran. Saat Anda menggunakan rantai peran, kredensial baru Anda dibatasi hingga durasi maksimum satu jam. Bila Anda menggunakan peran untuk memberikan izin ke aplikasi yang berjalan pada EC2 instance, aplikasi tersebut tidak tunduk pada batasan ini.

5. Berikan URL kepada pengguna atau panggil URL atas nama pengguna.

Titik akhir federasi URL yang disediakan berlaku selama 15 menit setelah dibuat. Ini berbeda dari durasi (dalam detik) sesi kredensi keamanan sementara yang terkait dengan. URL Kredensial tersebut berlaku selama durasi yang Anda tentukan saat membuatnya, dimulai sejak saat informasi tersebut dibuat.

> **Important**
>
> URLMemberikan akses ke AWS sumber daya Anda melalui AWS Management Console jika Anda telah mengaktifkan izin di kredenial keamanan sementara terkait. Untuk alasan ini, Anda harus memperlakukan URL sebagai rahasia. Sebaiknya kembalikan URL melalui pengalihan aman, misalnya, dengan menggunakan kode status HTTP respons 302 melalui koneksiSSL. Untuk informasi lebih lanjut tentang kode status HTTP respons 302, buka RFC2616, bagian 10.3.3.

Untuk menyelesaikan tugas-tugas ini, Anda dapat menggunakan HTTPSQuery API for AWS Identity and Access Management (IAM) dan AWS Security Token Service (AWS STS). Atau, Anda dapat menggunakan bahasa pemrograman, seperti Java, Ruby, atau C #, bersama dengan yang sesuai. AWS SDK Masing-masing metode ini dijelaskan dalam topik-topik berikut.

Topik

- Contoh kode menggunakan API operasi IAM kueri
- Contoh kode menggunakan Python
- Contoh kode menggunakan Java
- Contoh yang menunjukkan bagaimana membangun URL (Ruby)

Contoh kode menggunakan API operasi IAM kueri

Anda dapat membuat sebuah URL yang memberi pengguna federasi Anda akses langsung ke file. AWS Management Console Tugas ini menggunakan IAM dan AWS STS HTTPS QueryAPI. Untuk informasi selengkapnya tentang membuat permintaan pertanyaan, lihat Membuat Permintaan Kueri.

> **Note**
>
> Prosedur berikut berisi contoh string teks. Untuk meningkatkan keterbacaan, jeda baris telah ditambahkan ke beberapa contoh yang lebih panjang. Saat Anda membuat string ini untuk Anda gunakan sendiri, Anda harus mengurangi setiap pecahan baris.

Untuk memberikan akses pengguna federasi ke sumber daya Anda dari AWS Management Console

1. Autentikasi pengguna di sistem identitas dan otorisasi Anda.
2. Dapatkan kredensial keamanan sementara untuk pengguna. Kredensi sementara terdiri dari ID kunci akses, kunci akses rahasia, dan token sesi. Untuk informasi lebih lanjut tentang membuat kredensial sementara, lihat Kredensi keamanan sementara di IAM.

   Untuk mendapatkan kredensi sementara, Anda menelepon AWS STS AssumeRoleAPI(disarankan) atau. GetFederationTokenAPI Untuk informasi selengkapnya tentang perbedaan antara API operasi ini, lihat Memahami API Opsi untuk Mendelegasikan Akses ke AWS Akun Anda dengan Aman di Blog AWS Keamanan.

   > **Important**
   >
   > Bila Anda menggunakan GetFederationTokenAPIuntuk membuat kredenal keamanan sementara, Anda harus menentukan izin yang diberikan kredensialnya kepada pengguna yang mengambil peran tersebut. Untuk setiap API operasi yang dimulai`AssumeRole*`, Anda menggunakan IAM peran untuk menetapkan izin. Untuk API operasi lainnya, mekanismenya bervariasi denganAPI. Untuk detail selengkapnya, lihat Izin untuk kredensial keamanan sementara. Selain itu, jika Anda menggunakan `AssumeRole*` API operasi, Anda harus memanggil mereka sebagai IAM pengguna dengan kredensi jangka panjang. Jika tidak, panggilan ke titik akhir federasi pada langkah 3 gagal.
3. Setelah Anda mendapatkan kredensi keamanan sementara, buat kredenal tersebut menjadi string JSON sesi untuk menukarnya dengan token masuk. Contoh berikut menunjukkan cara mengkode kredensial. Anda mengganti teks placeholder dengan nilai yang sesuai dari kredensial yang Anda terima di langkah sebelumnya.

   ```
   {"sessionId":"*** temporary access key ID ***",
   "sessionKey":"*** temporary secret access key ***",
   "sessionToken":"*** session token ***"}
   ```

4. URLencode string sesi dari langkah sebelumnya. Karena informasi yang Anda encoding itu sensitif, kami menganjurkan agar Anda menghindari penggunaan layanan web untuk encoding ini. Alih-alih, gunakan fungsi atau fitur yang dipasang secara lokal di perangkat pengembangan Anda untuk mengenkode informasi ini dengan aman. Anda dapat menggunakan fungsi `urllib.quote_plus` di Python, fungsi `URLEncoder.encode` di Java, atau fungsi `CGI.escape` di Ruby. Lihat contoh-contoh ini nanti dalam topik ini.
5.

> **Note**
>
> AWS mendukung POST permintaan di sini.

Kirim permintaan Anda ke titik akhir AWS federasi:

`https://region-code.signin.aws.amazon.com/federation`

Untuk daftar kemungkinan *`region-code`* nilai, lihat kolom Region di titik akhir AWS Masuk. Anda dapat secara opsional menggunakan titik akhir federasi AWS Masuk default:

`https://signin.aws.amazon.com/federation`

Permintaan harus menyertakan `Action` dan `Session` parameter, dan (opsional) jika Anda menggunakan `AssumeRole*`APIoperasi, `SessionDuration` HTTP parameter seperti yang ditunjukkan pada contoh berikut.

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> Petunjuk berikut dalam langkah ini hanya berfungsi menggunakan GET permintaan.

`SessionDuration`HTTPParameter menentukan durasi sesi konsol. Ini terpisah dari durasi kredensial sementara yang Anda tentukan menggunakan parameter `DurationSeconds`. Anda dapat menentukan nilai maksimal `SessionDuration` adalah 43.200 (12 jam). Jika `SessionDuration` parameter hilang, maka sesi default ke durasi kredensional yang Anda ambil dari AWS STS langkah 2 (yang defaultnya satu jam). Lihat dokumentasi `AssumeRole` API

untuk rincian tentang cara menentukan durasi menggunakan `DurationSeconds` parameter. Kemampuan untuk membuat sesi konsol yang lebih lama dari satu jam bersifat intrinsik bagi `getSigninToken` operasi titik akhir federasi.

> **Note**
>
> - Jangan gunakan `SessionDuration` HTTP parameter jika Anda mendapatkan kredensi sementara dengan. `GetFederationToken` Melakukan hal itu akan menyebabkan kegagalan operasi.
> - Menggunakan kredensi untuk satu peran untuk mengambil peran yang berbeda disebut rantai peran. Saat Anda menggunakan rantai peran, kredensial baru Anda dibatasi hingga durasi maksimum satu jam. Bila Anda menggunakan peran untuk memberikan izin ke aplikasi yang berjalan pada EC2 instance, aplikasi tersebut tidak tunduk pada batasan ini.

Saat Anda mengaktifkan sesi konsol dengan durasi yang diperpanjang, Anda meningkatkan risiko eksposur kredensial. Untuk membantu mengurangi risiko ini, Anda dapat langsung menonaktifkan sesi konsol aktif untuk peran apa pun dengan memilih Cabut Sesi di halaman konsol Ringkasan Peran. IAM Untuk informasi selengkapnya, lihat Mencabut kredensi IAM keamanan sementara peran.

Berikut ini adalah contoh seperti apa permintaan Anda. Garis dibungkus di sini agar mudah dibaca, tetapi Anda harus mengirimkannya sebagai string satu baris.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

Tanggapan dari titik akhir federasi adalah JSON dokumen dengan `SigninToken` nilai. Ini akan terlihat serupa dengan contoh berikut.

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWS mendukung POST permintaan di sini.

Terakhir, buat URL yang dapat digunakan pengguna federasi Anda untuk mengakses file. AWS Management Console URLIni adalah URL titik akhir federasi yang sama dengan yang Anda gunakanStep 5, ditambah parameter berikut:

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> Instruksi berikut dalam langkah ini hanya berfungsi menggunakan GETAPI.

Contoh berikut menunjukkan seperti URL apa tampilan final. URLIni berlaku selama 15 menit dari saat dibuat. Kredensial keamanan sementara dan sesi konsol yang disematkan di URL dalamnya berlaku selama durasi yang Anda tentukan dalam `SessionDuration` HTTP parameter saat Anda memintanya terlebih dahulu.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
```

```
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Contoh kode menggunakan Python

Contoh berikut menunjukkan bagaimana menggunakan Python untuk secara terprogram membangun sebuah URL yang memberikan pengguna federasi akses langsung ke file. AWS Management Console Berikut ini adalah dua contoh:

- Federasi melalui GET permintaan untuk AWS
- Federasi melalui POST permintaan untuk AWS

Kedua contoh menggunakan AWS SDK for Python (Boto3)dan AssumeRoleAPIuntuk mendapatkan kredensi keamanan sementara.

Gunakan GET Permintaan

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your Akun AWS,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)
```

```
# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
print (request_url)
```

Gunakan POST Permintaan

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your A Akun AWS,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
```

```
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']

jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
        form.appendChild(hiddenField);
    }}
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)
```

```
driver = webdriver.Chrome()
driver.execute_script(jsrequest);
```

Contoh kode menggunakan Java

Contoh berikut menunjukkan bagaimana menggunakan Java untuk secara terprogram membangun sebuah URL yang memberikan pengguna federasi akses langsung ke. AWS Management Console Cuplikan kode berikut menggunakan AWS SDKuntuk Java.

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");
```

```
// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.

String policy = "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Action\":\"sns:*\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
                           "?Action=getSigninToken" +
                           "&DurationSeconds=43200" +
                           "&SessionType=json&Session=" +
                           URLEncoder.encode(sessionJson,"UTF-8");

URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
```

```
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

Contoh yang menunjukkan bagaimana membangun URL (Ruby)

Contoh berikut menunjukkan bagaimana menggunakan Ruby untuk secara terprogram membangun sebuah URL yang memberikan pengguna federasi akses langsung ke. AWS Management Console Cuplikan kode ini menggunakan AWS SDKuntuk Ruby.

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
```

```
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",\"Statement\":{\"Effect\":\"Allow\",\"Action\":
\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
                       "&SessionType=json&Session=" +
                       CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read

# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
```

```
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

Tag untuk AWS Identity and Access Management sumber daya

Tanda adalah label atribut khusus yang dapat Anda tetapkan ke sumber daya AWS . Setiap tag memiliki dua bagian:

- Sebuah kunci tag (misalnya, `CostCenter`, `Environment`, `Project`, atau `Purpose`).
- Bidang opsional yang dikenal sebagai nilai tag (misalnya, `111122223333`, `Production`, atau nama tim). Mengabaikan nilai tag sama dengan menggunakan rangkaian kosong.

Bersama-sama ini dikenal sebagai pasangan nilai-kunci. Untuk batasan jumlah tag yang dapat Anda miliki pada IAM sumber daya, lihatIAMdan AWS STS kuota.

> **Note**
>
> Untuk detail tentang sensitivitas huruf besar untuk kunci tag dan nilai kunci tag, lihatCase sensitivity.

Tag membantu Anda mengidentifikasi dan mengatur AWS sumber daya Anda. Banyak AWS layanan mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya terkait. Misalnya, Anda dapat menetapkan tag yang sama ke IAM peran yang ditetapkan ke bucket Amazon S3. Untuk informasi selengkapnya tentang strategi penandaan, lihat Panduan Pengguna AWS sumber daya Penandaan.

Selain mengidentifikasi, mengatur, dan melacak IAM sumber daya Anda dengan tag, Anda dapat menggunakan tag dalam IAM kebijakan untuk membantu mengontrol siapa yang dapat melihat dan berinteraksi dengan sumber daya Anda. Untuk mempelajari selengkapnya tentang penggunaan tanda untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lulus tag sesi di AWS STS.

Topik

Pilih konvensi penamaan AWS tag

Saat Anda mulai melampirkan tag ke IAM sumber daya Anda, pilih konvensi penamaan tag Anda dengan hati-hati. Terapkan konvensi yang sama untuk semua AWS tag Anda. Ini sangat penting jika Anda menggunakan tag dalam kebijakan untuk mengontrol akses ke AWS sumber daya. Jika Anda sudah menggunakan tag di AWS, tinjau konvensi penamaan Anda dan sesuaikan sesuai kebutuhan.

> **Note**
>
> Jika akun Anda adalah anggota AWS Organizations, lihat Kebijakan tag di panduan pengguna Organizations untuk mempelajari lebih lanjut tentang penggunaan tag di Organizations.

Praktik terbaik untuk penamaan tag

Ini adalah beberapa praktik terbaik dan konvensi penamaan untuk tag.

Pastikan bahwa nama tag digunakan secara konsisten. Misalnya, tag `CostCenter` dan `costcenter` berbeda, jadi satu mungkin dikonfigurasi sebagai tag alokasi biaya untuk analisis keuangan dan pelaporan dan yang lainnya mungkin tidak. Demikian pula, `Name` tag muncul di AWS Konsol untuk banyak sumber daya, tetapi `name` tag tidak. Untuk detail tentang sensitivitas huruf besar untuk kunci tag dan nilai kunci tag, lihatCase sensitivity.

Sejumlah tag telah ditentukan sebelumnya oleh AWS atau dibuat secara otomatis oleh berbagai AWS layanan. Banyak nama tag AWS yang ditentukan menggunakan semua huruf kecil, dengan tanda hubung yang memisahkan kata dalam nama, dan awalan untuk mengidentifikasi layanan sumber untuk tag tersebut. Sebagai contoh:

- `aws:ec2spot:fleet-request-id`mengidentifikasi Permintaan Instans EC2 Spot Amazon yang meluncurkan instance.
- `aws:cloudformation:stack-name`mengidentifikasi AWS CloudFormation tumpukan yang menciptakan sumber daya.
- `elasticbeanstalk:environment-name`mengidentifikasi aplikasi yang menciptakan sumber daya.

Pertimbangkan untuk memberi nama tag Anda menggunakan semua huruf kecil, dengan tanda hubung yang memisahkan kata, dan awalan yang mengidentifikasi nama organisasi atau nama yang disingkat. Misalnya, untuk perusahaan fiktif bernama AnyCompany, Anda dapat menentukan tag seperti:

- `anycompany:cost-center`untuk mengidentifikasi kode Pusat Biaya internal
- `anycompany:environment-type`untuk mengidentifikasi apakah lingkungan adalah pengembangan, pengujian, atau produksi
- `anycompany:application-id`untuk mengidentifikasi aplikasi sumber daya diciptakan untuk

Awalan memastikan bahwa tag diidentifikasi dengan jelas sebagai telah ditentukan oleh organisasi Anda dan bukan oleh AWS atau alat pihak ketiga yang mungkin Anda gunakan. Menggunakan semua huruf kecil dengan tanda hubung untuk pemisah menghindari kebingungan tentang cara menggunakan huruf besar pada nama tag. Misalnya, `anycompany:project-id` lebih mudah diingat daripada`ANYCOMPANY:ProjectID`,`anycompany:projectID`, atau`Anycompany:ProjectId`.

Aturan untuk menandai dan IAM AWS STS

Sejumlah konvensi mengatur pembuatan dan penerapan tag di IAM dan. AWS STS

Penamaan tag

Perhatikan konvensi berikut saat merumuskan konvensi penamaan tag untuk IAM sumber daya, sesi AWS STS peran asumsi, dan sesi pengguna gabungan: AWS STS

Persyaratan karakter — Kunci dan nilai tag dapat mencakup kombinasi huruf, angka, spasi, dan _.:/= + - @ simbol.

Sensitivitas kasus — Sensitivitas kasus untuk kunci tag berbeda tergantung pada jenis IAM sumber daya yang diberi tag. Nilai kunci tag untuk IAM pengguna dan peran tidak peka huruf besar/kecil, tetapi case dipertahankan. Ini berarti Anda tidak dapat memisahkan kunci tag **`Department`** dan **`department`**. Jika Anda sudah memberi tag ke pengguna dengan tag **`Department=finance`** dan Anda menambahkan tag **`department=hr`**, itu menggantikan tag pertama. Tag kedua tidak ditambahkan.

Untuk jenis IAM sumber daya lainnya, nilai kunci tag peka huruf besar/kecil. Ini berarti Anda dapat memisahkan kunci tanda **`Costcenter`** dan **`costcenter`**. Sebagai contoh, jika Anda telah menandai kebijakan yang dikelola pelanggan dengan tanda **`Costcenter = 1234`** dan Anda menambahkan tanda **`costcenter = 5678`**, kebijakan akan memiliki kunci tanda **`Costcenter`** dan **`costcenter`**.

Sebagai praktik terbaik, kami menyarankan agar Anda menghindari penggunaan tanda yang serupa dengan perlakuan huruf besar kecil yang tidak konsisten. Kami menyarankan Anda memutuskan strategi untuk memanfaatkan tag, dan secara konsisten menerapkan strategi itu di semua jenis sumber daya. Untuk mempelajari lebih lanjut tentang praktik terbaik untuk penandaan, lihat Menandai AWS Sumber Daya di. Referensi Umum AWS

Daftar berikut menunjukkan perbedaan sensitivitas kasus untuk kunci tag yang dilampirkan ke IAM sumber daya.

Nilai kunci tanda tidak peka huruf besar kecil:

- IAMperan
- IAMpengguna

Nilai kunci tanda peka huruf besar kecil:

- Kebijakan yang dikelola pelanggan
- Profil instans
- Penyedia identitas OpenID Connect
- SAMLpenyedia identitas
- Sertifikat server
- MFAPerangkat virtual

Selain itu, aturan berikut ini berlaku:

- Anda tidak dapat membuat kunci tanda atau nilai yang dimulai dengan teks **`aws:`**. Awalan tag ini dicadangkan untuk penggunaan AWS internal.
- Anda dapat membuat tanda dengan nilai kosong seperti**`phoneNumber =`** . Anda tidak dapat membuat kunci tanda kosong.
- Anda tidak dapat menentukan beberapa nilai dalam satu tag, tetapi Anda dapat membuat struktur multinilai khusus dalam satu nilai. Misalnya, anggap bahwa pengguna Zhang bekerja di tim engineer dan tim QA. Jika Anda melampirkan atnda **`team = Engineering`** lalu melampirkan tanda **`team = QA`**, Anda mengubah nilai dari tanda dari **`Engineering`** ke **`QA`**. Sebaliknya, Anda dapat menyertakan beberapa nilai dalam satu tanda dengan pemisah kustom. Dalam contoh ini, Anda dapat memberikan tag **`team = Engineering:QA`** ke Zhang.

> **Note**
>
> Untuk mengontrol akses ke engineer dalam contoh ini menggunakan tanda **`team`**, Anda harus membuat kebijakan yang memungkinkan setiap konfigurasi yang mungkin mencakup **`Engineering`**, termasuk **`Engineering:QA`**. Untuk mem-pelajari selengkapnya tentang penggunaan tanda dalam kebijakan, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Menerapkan dan mengubah tag

Perhatikan konvensi berikut saat melampirkan tag ke IAM sumber daya:

- Anda dapat menandai sebagian besar IAM sumber daya, tetapi bukan grup, peran yang diasumsikan, laporan akses, atau perangkat berbasis perangkat kerasMFA.

- Anda tidak dapat menggunakan Editor Tag untuk menandai IAM sumber daya. Editor Tag tidak mendukung IAM tag. Untuk informasi tentang penggunaan Editor Tanda dengan layanan lain, lihat Bekerja dengan Editor Tanda dalam Panduan Pengguna AWS Resource Groups .
- Untuk menandai IAM sumber daya, Anda harus memiliki izin khusus. Untuk memberi atau menghapus tanda sumber daya, Anda juga harus memiliki izin untuk mencantumkan tanda. Untuk informasi selengkapnya, lihat daftar topik untuk setiap IAM sumber daya di akhir halaman ini.
- Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.
- Anda dapat menerapkan tag yang sama ke beberapa IAM sumber daya. Misalnya, Anda memiliki departemen bernama `AWS_Development` dengan 12 anggota. Anda bisa memiliki 12 pengguna dan satu peran dengan kunci tag **`department`** dan nilai **`awsDevelopment`** (**`department = awsDevelopment`**). Anda juga dapat menggunakan tag yang sama pada sumber daya di layanan yang mendukung pemberian tag.
- IAMentitas (pengguna atau peran) tidak dapat memiliki beberapa instance dari kunci tag yang sama. Misalnya, jika Anda memiliki pengguna dengan pasangan nilai kunci tag**`costCenter = 1234`**, Anda kemudian dapat melampirkan pasangan nilai kunci tag. **`costCenter = 5678`** IAMmemperbarui nilai **`costCenter`** tag ke**`5678`**.
- Untuk mengedit tag yang dilampirkan ke IAM entitas (pengguna atau peran), lampirkan tag dengan nilai baru untuk menimpa tag yang ada. Sebagai contoh, anggap Anda memiliki pengguna dengan pasangan nilai-kunci tanda **`department = Engineering`**. Jika perlu memindahkan pengguna ke departemen QA, Anda dapat melampirkan pasangan nilai-kunci tanda **`department = QA`** ke pengguna. Ini menghasilkan nilai **`Engineering`** dari kunci tag **`department`** yang digantikan dengan nilai **`QA`**.

Tag IAM pengguna

Anda dapat menggunakan pasangan nilai kunci IAM tag untuk menambahkan atribut khusus ke pengguna. IAM Misalnya, untuk menambahkan informasi lokasi ke pengguna, Anda dapat menambahkan kunci tag **`location`** dan nilai tanda **`us_wa_seattle`**. Atau Anda bisa menggunakan tiga pasangan kunci nilai tanda lokasi yang terpisah: **`loc-country = us`**, **`loc-state = wa`**, dan **`loc-city = seattle`**. Anda dapat menggunakan tanda untuk mengontrol akses pengguna ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke pengguna. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lulus tag sesi di AWS STS.

Izin yang diperlukan untuk menandai pengguna IAM

Anda harus mengonfigurasi izin untuk memungkinkan IAM pengguna menandai pengguna lain. Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Untuk memungkinkan IAM pengguna menambahkan, membuat daftar, atau menghapus tag untuk pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM pengguna yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<username>* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Untuk memungkinkan IAM pengguna mengelola sendiri tag

Tambahkan pernyataan berikut ini ke kebijakan izin untuk pengguna agar pengguna dapat mengelola tag mereka sendiri. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
```

```
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

Untuk memungkinkan IAM pengguna menambahkan tag ke pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM pengguna yang perlu menambahkan, tetapi tidak menghapus, tag untuk pengguna tertentu.

> **Note**
>
> Tindakan `iam:TagUser` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListUserTags`

Untuk menggunakan kebijakan ini, ganti *<username>* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tanda pada pengguna IAM (konsol)

Anda dapat mengelola tag untuk IAM pengguna dari file AWS Management Console.

Untuk mengelola tanda pada pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran lalu pilih nama pengguna yang ingin Anda edit.
3. Pilih Tag kemudian selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tag baru jika pengguna belum memiliki tag.
   - Pilih Kelola tag untuk mengelola kumpulan tag yang ada.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Lalu pilih Simpan Perubahan.

Mengelola tag pada IAM pengguna (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk IAM pengguna. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk IAM pengguna.

Untuk mencantumkan tag yang saat ini dilampirkan ke IAM pengguna (AWS CLI atau AWS API)

- AWS CLI: aws iam list-user-tags
- AWS API: ListUserTags

Untuk melampirkan tag ke IAM pengguna (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-user
- AWS API: TagUser

Untuk menghapus tag dari IAM pengguna (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-user
- AWS API: UntagUser

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

IAMPeran tag

Anda dapat menggunakan pasangan nilai kunci IAM tag untuk menambahkan atribut khusus ke peranIAM. Misalnya, untuk menambahkan informasi lokasi ke peran, Anda dapat menambahkan kunci tag **`location`** dan nilai tanda **`us_wa_seattle`**. Atau Anda bisa menggunakan tiga pasangan kunci nilai tanda lokasi yang terpisah: **`loc-country = us`**, **`loc-state = wa`**, dan **`loc-city = seattle`**. Anda dapat menggunakan tanda untuk mengontrol akses peran ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke peran. Untuk mempelajari selengkapnya tentang penggunaan tanda untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lulus tag sesi di AWS STS.

Izin yang diperlukan untuk menandai peran IAM

Anda harus mengonfigurasi izin untuk mengizinkan IAM peran menandai entitas lain (pengguna atau peran). Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListRoleTags`
- `iam:TagRole`
- `iam:UntagRole`
- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Untuk mengizinkan IAM peran menambah, mencantumkan, atau menghapus tag untuk pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM peran yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<username>`* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
```

```
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Untuk mengizinkan IAM peran menambahkan tag ke pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM peran yang perlu menambahkan, tetapi tidak menghapus, tag untuk pengguna tertentu.

Untuk menggunakan kebijakan ini, ganti *<username>* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Untuk mengizinkan IAM peran menambah, mencantumkan, atau menghapus tag untuk peran tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM peran yang perlu mengelola tag. Ganti *<rolename>* dengan nama peran yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tanda di peran IAM (konsol)

Anda dapat mengelola tag untuk IAM peran dari AWS Management Console.

Untuk mengelola tanda pada peran (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran lalu pilih nama peran yang ingin Anda edit.
3. Pilih Tag kemudian selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tag baru jika peran belum memiliki tag.
   - Pilih Kelola tag untuk mengelola kumpulan tag yang ada.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Kemudian, pilih Simpan perubahan.

Mengelola tag pada IAM peran (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk IAM peran. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk IAM peran.

Untuk mencantumkan tag yang saat ini dilampirkan ke IAM peran (AWS CLI atau AWS API)

- AWS CLI: aws iam list-role-tags
- AWS API: ListRoleTags

Untuk melampirkan tag ke IAM peran (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-role
- AWS API: TagRole

Untuk menghapus tag dari IAM peran (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-role

- AWS API: UntagRole

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

Tandai kebijakan yang dikelola pelanggan

Anda dapat menggunakan pasangan nilai kunci IAM tag untuk menambahkan atribut kustom ke kebijakan terkelola pelanggan Anda. Misalnya, untuk menandai kebijakan dengan informasi departemen, Anda dapat menambahkan kunci tanda **`Department`** dan nilai tanda **`eng`**. Atau, Anda mungkin ingin menandai kebijakan guna menunjukkan bahwa kebijakan tersebut adalah untuk lingkungan tertentu, seperti **`Environment = lab`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke identitas. Untuk mempelajari selengkapnya tentang penggunaan tanda untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lulus tag sesi di AWS STS.

Izin yang diperlukan untuk menandai kebijakan terkelola pelanggan

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) menandai kebijakan yang dikelola pelanggan. Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListPolicyTags`
- `iam:TagPolicy`
- `iam:UntagPolicy`

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan, mencantumkan, atau menghapus tag untuk kebijakan yang dikelola pelanggan

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<policyname>`* dengan nama kebijakan yang tagnya

perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan tag ke kebijakan terkelola pelanggan tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menambahkan, tetapi tidak menghapus, tag untuk kebijakan tertentu.

> **Note**
>
> Tindakan `iam:TagPolicy` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListPolicyTags`

Untuk menggunakan kebijakan ini, ganti *<policyname>* dengan nama kebijakan yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tag pada kebijakan yang dikelola IAM pelanggan (konsol)

Anda dapat mengelola tag untuk kebijakan yang dikelola IAM pelanggan dari AWS Management Console.

Untuk mengelola tanda pada kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Kebijakan lalu pilih nama kebijakan terkelola pelanggan yang ingin Anda edit.
3. Pilih tab Tag dan kemudian pilih Kelola tag.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Lalu pilih Simpan Perubahan.

Mengelola tag pada kebijakan yang dikelola IAM pelanggan (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk kebijakan yang dikelola IAM pelanggan. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk kebijakan yang dikelola IAM pelanggan.

Untuk mencantumkan tag yang saat ini dilampirkan ke kebijakan yang dikelola IAM pelanggan (AWS CLI atau AWS API)

- AWS CLI: aws iam list-policy-tags
- AWS API: ListPolicyTags

Untuk melampirkan tag ke kebijakan yang dikelola IAM pelanggan (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-policy
- AWS API: TagPolicy

Untuk menghapus tag dari kebijakan yang dikelola IAM pelanggan (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-policy
- AWS API: UntagPolicy

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

Tandai OpenID Connect (OIDC) penyedia identitas

Anda dapat menggunakan nilai kunci IAM tag untuk menambahkan atribut kustom ke penyedia identitas OpenID IAM Connect ()OIDC. Misalnya, untuk mengidentifikasi penyedia OIDC identitas, Anda dapat menambahkan kunci tag **`google`** dan nilai tag**`oidc`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke objek. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Izin diperlukan untuk menandai penyedia identitas IAM OIDC

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) menandai penyedia IAM OIDC identitas. Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

Untuk mengizinkan IAM entitas menambah, mencantumkan, atau menghapus tag untuk penyedia IAM OIDC identitas

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<OIDCProviderName>`* dengan nama OIDC penyedia yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
        "iam:UntagOpenIDConnectProvider"
    ],
```

```
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan tag ke penyedia IAM OIDC identitas tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menambahkan, tetapi tidak menghapus, tag untuk penyedia identitas tertentu.

> **Note**
>
> Tindakan `iam:TagOpenIDConnectProvider` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListOpenIDConnectProviderTags`

Untuk menggunakan kebijakan ini, ganti *<OIDCProviderName>* dengan nama OIDC penyedia yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tag pada penyedia IAM OIDC identitas (konsol)

Anda dapat mengelola tag untuk penyedia IAM OIDC identitas dari AWS Management Console.

Untuk mengelola tag pada penyedia OIDC identitas (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Penyedia identitas, kemudian pilih nama penyedia identitas yang ingin Anda edit.

3. Pilih tab Tag, lalu di bagian Tag, pilih Kelola tag dan kemudian selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tag jika penyedia OIDC identitas belum memiliki tag atau menambahkan tag baru.
   - Mengedit kunci dan nilai tanda yang ada.
   - Untuk menghapus tanda, pilih Hapus tanda.
4. Lalu pilih Simpan Perubahan.

Mengelola tag pada penyedia IAM OIDC identitas (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk penyedia IAM OIDC identitas. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk penyedia IAM OIDC identitas.

Untuk mencantumkan tag yang saat ini dilampirkan ke penyedia IAM OIDC identitas (AWS CLI atau AWS API)

- AWS CLI: aws iam list-open-id-connect -provider-tag
- AWS API: ListOpenIDConnectProviderTags

Untuk melampirkan tag ke penyedia IAM OIDC identitas (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-open-id-connect -penyedia
- AWS API: TagOpenIDConnectProvider

Untuk menghapus tag dari penyedia IAM OIDC identitas (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-open-id-connect -penyedia
- AWS API: UntagOpenIDConnectProvider

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

Tag penyedia IAM SAML identitas

Anda dapat menggunakan pasangan nilai kunci IAM tag untuk menambahkan atribut khusus ke penyedia SAML identitas. Misalnya, untuk mengidentifikasi penyedia, Anda dapat menambahkan kunci tanda **`okta`** dan nilai tanda **`saml`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke objek. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Izin diperlukan untuk menandai penyedia identitas SAML

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) menandai Penyedia Identitas SAML berbasis 2.0 (). IdPs Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

Untuk mengizinkan IAM entitas (pengguna atau peran) menambah, mencantumkan, atau menghapus tag untuk penyedia SAML identitas

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<SAMLProviderName>* dengan nama SAML penyedia yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan tag ke penyedia SAML identitas tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menambahkan, tetapi tidak menghapus, tag untuk SAML penyedia tertentu.

> **Note**
>
> Tindakan `iam:TagSAMLProvider` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListSAMLProviderTags`

Untuk menggunakan kebijakan ini, ganti *<SAMLProviderName>* dengan nama SAML penyedia yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tag pada penyedia IAM SAML identitas (konsol)

Anda dapat mengelola tag untuk Penyedia IAM SAML Identitas dari AWS Management Console.

Untuk mengelola tag pada penyedia SAML identitas (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Penyedia identitas lalu pilih nama penyedia SAML identitas yang ingin Anda edit.
3. Pilih tab Tag, lalu di bagian Tag, pilih Kelola tag dan kemudian selesaikan salah satu tindakan berikut:

    - Pilih Tambahkan tag jika penyedia SAML identitas belum memiliki tag atau menambahkan tag baru.

- Mengedit kunci dan nilai tanda yang ada.
- Untuk menghapus tanda, pilih Hapus tanda.

4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Lalu pilih Simpan Perubahan.

Mengelola tag pada penyedia IAM SAML identitas (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk penyedia IAM SAML identitas. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk penyedia IAM SAML identitas.

Untuk mencantumkan tag yang saat ini dilampirkan ke penyedia SAML identitas (AWS CLI atau AWS API)

- AWS CLI: aws iam list-saml-provider-tags
- AWS API: L istSAMLProvider Tag

Untuk melampirkan tag ke penyedia SAML identitas (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-saml-provider
- AWS API: T agSAMLProvider

Untuk menghapus tag dari penyedia SAML identitas (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-saml-provider
- AWS API: U ntagSAMLProvider

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

Menandai profil instance untuk EC2 peran Amazon

Saat meluncurkan EC2 instans Amazon, Anda menentukan IAM peran yang akan dikaitkan dengan instance tersebut. Profil instance adalah wadah untuk IAM peran yang dapat Anda gunakan untuk

meneruskan informasi peran ke EC2 instans Amazon saat instance dimulai. Anda dapat menandai profil instance saat Anda menggunakan AWS CLI atau AWS API.

Anda dapat menggunakan pasangan nilai kunci IAM tag untuk menambahkan atribut khusus ke profil instance. Misalnya, untuk menambahkan informasi departemen ke profil instans, Anda dapat menambahkan kunci tanda **`access-team`** dan nilai tanda **`eng`**. Melakukan hal ini memberikan prinsipal dengan akses tanda yang sesuai ke profil instans dengan tanda yang sama. Anda bisa menggunakan beberapa tag pasangan kunci-nilai tanda untuk menentukan tim dan proyek: **`access-team = eng`** , dan **`project = peg`**. Anda dapat menggunakan tanda untuk mengontrol akses pengguna ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke pengguna. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lulus tag sesi di AWS STS.

Izin yang diperlukan untuk menandai profil instans

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) menandai profil instance. Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListInstanceProfileTags`
- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan, mencantumkan, atau menghapus tag untuk profil instance

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<InstanceProfileName>* dengan nama profil instance yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
```

```
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan tag ke profil instance tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menambahkan, tetapi tidak menghapus, tag untuk profil instance tertentu.

> **Note**
>
> Tindakan `iam:TagInstanceProfile` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListInstanceProfileTags`

Untuk menggunakan kebijakan ini, ganti *<InstanceProfileName>* dengan nama profil instance yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tag pada profil contoh (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk profil instans. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk profil misalnya.

Untuk mencantumkan tag yang saat ini dilampirkan ke profil instance (AWS CLI atau AWS API)

- AWS CLI: aws iam list-instance-profile-tags
- AWS API: ListInstanceProfileTags

Untuk melampirkan tag ke profil instance (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-instance-profile
- AWS API: TagInstanceProfile

Untuk menghapus tag dari profil instance (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-instance-profile
- AWS API: UntagInstanceProfile

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

Tandai sertifikat server

Jika Anda menggunakan IAM untuk mengelolaSSL/TLSsertifikat, Anda dapat menandai sertifikat server dalam IAM menggunakan AWS CLI atau AWS API. Untuk sertifikat di Wilayah yang didukung oleh AWS Certificate Manager (ACM), sebaiknya Anda menggunakan, ACM bukan IAM untuk menyediakan, mengelola, dan menerapkan sertifikat server Anda. Di Wilayah yang tidak didukung, Anda harus menggunakan IAM sebagai manajer sertifikat. Untuk mempelajari Wilayah mana yang ACM mendukung, lihat AWS Certificate Manager titik akhir dan kuota di. Referensi Umum AWS

Anda dapat menggunakan pasangan nilai kunci IAM tag untuk menambahkan atribut khusus ke sertifikat server. Misalnya, untuk menambahkan informasi tentang pemilik atau administrator sertifikat server, tambahkan kunci tanda **`owner`** dan nilai tanda **`net-eng`**. Atau Anda dapat menentukan pusat biaya dengan menambahkan kunci tanda **`CostCenter`** dan nilai tanda **`1234`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke sumber daya. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lulus tag sesi di AWS STS.

Izin yang diperlukan untuk menandai sertifikat server

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) menandai sertifikat server. Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

Untuk mengizinkan IAM entitas (pengguna atau peran) menambah, mencantumkan, atau menghapus tag untuk sertifikat server

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<CertificateName>* dengan nama sertifikat server yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan tag ke sertifikat server tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menambahkan, tetapi tidak menghapus, tag untuk sertifikat server tertentu.

> **Note**
>
> Tindakan `iam:TagServerCertificate` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListServerCertificateTags`

Untuk menggunakan kebijakan ini, ganti *<CertificateName>* dengan nama sertifikat server yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tag pada sertifikat server (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk sertifikat server. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk sertifikat server.

Untuk mencantumkan tag yang saat ini dilampirkan ke sertifikat server (AWS CLI atau AWS API)

- AWS CLI: aws iam list-server-certificate-tags
- AWS API: ListServerCertificateTags

Untuk melampirkan tag ke sertifikat server (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-server-certificate
- AWS API: TagServerCertificate

Untuk menghapus tag dari sertifikat server (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-server-certificate
- AWS API: UntagServerCertificate

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

Tandai MFA perangkat virtual

Anda dapat menggunakan pasangan nilai kunci IAM tag untuk menambahkan atribut khusus ke perangkat virtualMFA. Misalnya, untuk menambahkan informasi pusat biaya untuk MFA perangkat virtual pengguna, Anda dapat menambahkan kunci tag **`CostCenter`** dan nilai tag**`1234`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke objek. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lulus tag sesi di AWS STS.

Izin diperlukan untuk menandai perangkat virtual MFA

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (pengguna atau peran) menandai MFA perangkat virtual. Anda dapat menentukan satu atau semua tindakan IAM tag berikut dalam IAM kebijakan:

- `iam:ListMFADeviceTags`
- `iam:TagMFADevice`
- `iam:UntagMFADevice`

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan, membuat daftar, atau menghapus tag untuk MFA perangkat virtual

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<MFATokenID>`* dengan nama MFA perangkat virtual yang

tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Untuk mengizinkan IAM entitas (pengguna atau peran) menambahkan tag ke MFA perangkat virtual tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk IAM entitas yang perlu menambahkan, tetapi tidak menghapus, tag untuk MFA perangkat tertentu.

> **Note**
>
> Tindakan `iam:TagMFADevice` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListMFADeviceTags`

Untuk menggunakan kebijakan ini, ganti *<MFATokenID>* dengan nama MFA perangkat virtual yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccessuntuk menyediakan akses penuh keIAM.

Mengelola tag pada MFA perangkat virtual (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk MFA perangkat virtual. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk MFA perangkat virtual.

Untuk mencantumkan tag yang saat ini dilampirkan ke MFA perangkat virtual (AWS CLI atau AWS API)

- AWS CLI: aws iam list-mfa-device-tags
- AWS API: L istMFADevice Tag

Untuk melampirkan tag ke MFA perangkat virtual (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-mfa-device
- AWS API: T agMFADevice

Untuk menghapus tag dari MFA perangkat virtual (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-mfa-device
- AWS API: U ntagMFADevice

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang penggunaan tag untuk menyetel izin terperinci lainnya dengan kebijakan IAM izin, lihat. Elemen kebijakan IAM: Variabel dan tanda

Lulus tag sesi di AWS STS

Tag sesi adalah atribut pasangan nilai kunci yang Anda lewati saat Anda mengambil IAM peran atau menyatukan pengguna. AWS STS Anda melakukan ini dengan membuat AWS CLI atau AWS API meminta melalui AWS STS atau melalui penyedia identitas Anda (IDP). Saat Anda menggunakan AWS STS untuk meminta kredensi keamanan sementara, Anda membuat sesi. Sesi berakhir dan memiliki kredensial, seperti pasangan access key dan token sesi. Saat Anda menggunakan kredensial sesi untuk membuat permintaan berikutnya, konteks permintaan mencakup kunci konteks `aws:PrincipalTag`. Anda dapat menggunakan kunci `aws:PrincipalTag` dalam elemen `Condition` elemen kebijakan Anda untuk mengizinkan atau menolak akses berdasarkan tag tersebut.

Saat Anda menggunakan kredensial sementara untuk membuat permintaan, prinsipal Anda mungkin mencakup serangkaian tag. Tag ini berasal dari sumber berikut:

1. Tag sesi — Tag diteruskan saat Anda mengambil peran atau menyatukan pengguna menggunakan AWS CLI atau AWS API. Untuk informasi selengkapnya tentang operasi ini, lihat Operasi pemberian tag ke sesi.
2. Tag sesi transitif masuk — Tag yang diwarisi dari sesi sebelumnya dalam rantai peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi dalam topik ini.
3. IAMtag — Tag yang dilampirkan pada peran yang Anda IAM asumsikan.

Topik

Operasi pemberian tag ke sesi

Anda dapat meneruskan tag sesi menggunakan yang berikut AWS CLI atau AWS API operasi di AWS STS. Fitur AWS Management Console Switch Role tidak memungkinkan Anda untuk meneruskan tag sesi.

Anda juga dapat mengatur tag sesi sebagai transitif. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Tabel berikut membandingkan metode untuk melewati tag sesi.

Operasi	Siapa yang bisa mengambil peran	Metode untuk melewati tag	Metode untuk mengatur tag transitif
assume-role CLIatau AssumeRole APIoperasi	Pengguna atau sesi IAM	TagsAPIparameter atau --tags CLI opsi	TransitiveTagKeys APIparameter atau --transitive-tag-keys CLI opsi
assume-role-with-saml CLIatau AssumeRoleWithSAML APIoperasi	Setiap pengguna yang diautentikasi menggunakan penyedia SAML identitas	Principal Tag SAMLatribut	TransitiveTagKeys SAMLAtribut
assume-role-with-web-identity CLIatau AssumeRoleWithWebIdentity APIoperasi	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	Principal Tag OIDCtoken	TransitiveTagKeys OIDCtoken
get-federation-token CLIatau GetFederationToken APIoperasi	IAMpengguna atau pengguna root	TagsAPIparameter atau --tags CLI opsi	Tidak didukung

Operasi yang mendukung penandaan sesi dapat gagal dalam kondisi berikut:

- Anda meneruskan lebih dari 50 tag sesi.

- Plaintext kunci tag sesi Anda melebihi 128 karakter.
- Plaintext dari nilai tag sesi Anda melebihi 256 karakter.
- Ukuran total plaintext kebijakan sesi melebihi 2048 karakter.
- Ukuran paket total dari kebijakan sesi gabungan dan tag terlalu besar. Jika operasi gagal, pesan kesalahan menunjukkan seberapa dekat kebijakan dan tag yang digabungkan mencapai batas ukuran atas, berdasarkan persentase.

Hal yang perlu diketahui tentang tag sesi

Sebelum menggunakan tag sesi, tinjau detail berikut tentang sesi dan tag.

- Saat menggunakan tag sesi, kebijakan kepercayaan untuk semua peran yang terhubung ke tag penerusan penyedia identitas (IDP) harus memiliki izin. `sts:TagSession` Untuk peran tanpa izin ini dalam kebijakan kepercayaan, `AssumeRole` operasi gagal.
- Saat Anda meminta sesi, Anda dapat menentukan tag utama sebagai tag sesi. Tag berlaku untuk permintaan yang Anda buat dengan menggunakan kredensial sesi.
- Tag sesi menggunakan pasangan kunci-nilai. Misalnya, untuk menambahkan informasi kontak ke sesi, Anda dapat menambahkan kunci tag sesi `email` dan nilai tag `johndoe@example.com`.
- Tag sesi harus mengikuti aturan untuk penamaan tag di IAM dan AWS STS. Topik ini mencakup informasi tentang kepekaan huruf besar-kecil dan awalan terbatas yang berlaku untuk tag sesi Anda.
- Tag sesi baru mengganti peran yang diasumsikan atau tag pengguna gabungan yang ada dengan kunci tag yang sama, terlepas dari kasus karakternya.
- Anda tidak dapat meneruskan tag sesi menggunakan AWS Management Console.
- Tag sesi hanya valid untuk sesi saat ini.
- Tag sesi mendukung perangkaian peran. Secara default, AWS STS tidak meneruskan tag ke sesi peran berikutnya. Anda juga dapat mengatur tag sesi sebagai transitif. Tag transitif bertahan selama rantai peran dan menggantikan `ResourceTag` nilai yang cocok setelah evaluasi kebijakan kepercayaan peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.
- Anda dapat menggunakan tag sesi untuk mengontrol akses ke sumber daya atau untuk mengontrol tag apa yang dapat diteruskan ke sesi selanjutnya. Untuk informasi selengkapnya, lihat IAMtutorial: Gunakan tag SAML sesi untuk ABAC.
- Anda dapat melihat tag utama untuk sesi Anda, termasuk tag sesi, di AWS CloudTrail log. Untuk informasi selengkapnya, lihat Melihat tag sesi di CloudTrail.

- Anda harus memberikan satu nilai untuk setiap tag sesi. AWS STS tidak mendukung tag sesi multi-nilai.
- Anda dapat meneruskan maksimum 50 tag sesi. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.
- AWS Konversi memampatkan kebijakan sesi yang diteruskan dan tag sesi yang digabungkan ke dalam format biner yang dikemas dengan batas terpisah. Jika Anda melebihi batas ini, pesan AWS API kesalahan AWS CLI atau menunjukkan seberapa dekat kebijakan dan tag yang digabungkan mencapai batas ukuran atas, berdasarkan persentase.

Izin diperlukan untuk menambahkan tag sesi

Selain tindakan yang cocok dengan API operasi, Anda harus memiliki tindakan khusus izin berikut dalam kebijakan Anda:

```
sts:TagSession
```

> **Important**
>
> Saat menggunakan tag sesi, kebijakan kepercayaan peran untuk semua peran yang terhubung ke penyedia identitas (IdP) harus memiliki izin `sts:TagSession`. `AssumeRole`Operasi gagal untuk peran apa pun yang terhubung ke tag sesi lewat IDP tanpa izin ini. Jika Anda tidak ingin memperbarui kebijakan kepercayaan peran untuk setiap peran, Anda dapat menggunakan instans IdP terpisah untuk meneruskan tag sesi. Kemudian, tambahkan `sts:TagSession` izin hanya untuk peran yang terhubung ke iDP terpisah.

Anda dapat menggunakan tindakan `sts:TagSession` dengan kunci syarat berikut.

- `aws:PrincipalTag`— Membandingkan tag yang dilampirkan pada prinsipal yang membuat permintaan dengan tag yang Anda tentukan dalam kebijakan. Misalnya, Anda dapat mengizinkan prinsipal untuk meneruskan tag sesi hanya jika prinsipal yang mengajukan permintaan memiliki tag yang ditentukan.
- `aws:RequestTag`— Membandingkan pasangan nilai kunci tag yang diteruskan dalam permintaan dengan pasangan tag yang Anda tentukan dalam kebijakan. Misalnya, Anda dapat mengizinkan prinsipal untuk meneruskan tag sesi yang ditentukan, tetapi hanya dengan nilai yang ditentukan.

- `aws:ResourceTag`— Membandingkan pasangan nilai kunci tag yang Anda tentukan dalam kebijakan dengan pasangan nilai kunci yang dilampirkan ke sumber daya. Misalnya, Anda dapat mengizinkan prinsipal untuk meneruskan tag sesi hanya jika peran yang mereka ambil menyertakan tag yang ditentukan.
- `aws:TagKeys`— Membandingkan kunci tag dalam permintaan dengan kunci yang Anda tentukan dalam kebijakan. Misalnya, Anda dapat mengizinkan prinsipal untuk hanya meneruskan tag sesi dengan kunci tag yang ditentukan. Kunci persyaratan ini membatasi rangkaian tag sesi maksimum yang dapat diteruskan.
- `sts:TransitiveTagKeys`- Membandingkan kunci tag sesi transitif dalam permintaan dengan yang ditentukan dalam kebijakan. Misalnya, Anda dapat menyusun kebijakan untuk mengizinkan prinsipal mengatur hanya tag tertentu sebagai transitif. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Misalnya, kebijakan kepercayaan peran berikut memungkinkan `test-session-tags` pengguna untuk mengambil peran dengan kebijakan terlampir. Ketika pengguna tersebut mengambil peran, mereka harus menggunakan AWS CLI atau AWS API untuk meneruskan tiga tag sesi yang diperlukan dan ID eksternal yang diperlukan. Selain itu, pengguna dapat memilih untuk mengatur tag `Project` dan `Department` tag sebagai transitif.

Example Contoh kebijakan kepercayaan peran untuk tag sesi

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
            }
        },
        {
```

```
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
                    ]
                }
            }
        }
    ]
}
```

Apa yang dilakukan kebijakan ini?

- `AllowIamUserAssumeRole`Pernyataan tersebut memungkinkan `test-session-tags` pengguna untuk mengambil peran dengan kebijakan terlampir. Ketika pengguna tersebut mengasumsikan peran tersebut, mereka harus meneruskan tag sesi yang diperlukan dan ID eksternal.
  - Blok syarat pertama dari pernyataan ini mengharuskan pengguna untuk meneruskan `Project`, `CostCenter`, dan `Department` tag sesi. Nilai tag tidak penting dalam pernyataan ini, sehingga Anda dapat menggunakan wildcard (*) untuk nilai tag. Blok ini memastikan bahwa pengguna melewati setidaknya tiga tag sesi ini. Jika tidak, operasi gagal. Pengguna dapat meneruskan tag tambahan.
  - Blok syarat kedua mengharuskan pengguna untuk meneruskan ID eksternal dengan nilai `Example987`.

- Pernyataan `AllowPassSessionTagsAndTransitive` memperbolehkan tindakan khusus izin `sts:TagSession`. Tindakan ini harus diperbolehkan sebelum pengguna dapat meneruskan tag sesi. Jika kebijakan Anda mencakup pernyataan pertama tanpa pernyataan kedua, pengguna tidak dapat mengasumsikan peran tersebut.
  - Blok syarat pertama dari pernyataan ini memperbolehkan pengguna untuk meneruskan nilai apa pun untuk sesi tag `CostCenter` dan `Project`. Anda melakukannya dengan menggunakan wildcard (*) untuk nilai tag dalam kebijakan, yang mengharuskan Anda menggunakan operator StringLikekondisi.
  - Blok syarat kedua memperbolehkan pengguna untuk hanya meneruskan `Engineering` atau `Marketing` nilai untuk tag sesi `Department`.
  - Blok kondisi ketiga mencantumkan kumpulan tag maksimum yang dapat Anda atur sebagai transitif. Pengguna dapat memilih untuk mengatur subset atau tidak ada tag sebagai transitif. Mereka tidak dapat menetapkan tag tambahan sebagai transitif. Anda dapat meminta agar mereka mengatur setidaknya salah satu tag sebagai transtitif dengan menambahkan blok syarat lain yang mencakup `"Null":{"sts:TransitiveTagKeys":"false"}`.

Meneruskan tag sesi dengan menggunakan AssumeRole

`AssumeRole`Operasi mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Anda dapat menggunakan kredensi IAM pengguna atau peran untuk menelepon. `AssumeRole` Untuk meneruskan tag sesi sambil mengambil peran, gunakan `--tags` AWS CLI opsi atau `Tags` AWS API parameter.

Untuk mengatur tag sebagai transitif, gunakan `--transitive-tag-keys` AWS CLI opsi atau `TransitiveTagKeys` AWS API parameter. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Contoh berikut menunjukkan permintaan sampel yang menggunakan `AssumeRole`. Dalam contoh ini, saat mengambil peran `my-role-example`, Anda membuat sesi bernama `my-session`. Anda menambahkan pasangan nilai kunci tanda sesi `Project` = `Automation`, `CostCenter` = `12345`, dan `Department` = `Engineering`. Anda juga mengatur tanda `Project` dan `Department` sebagai transitif dengan menentukan kuncinya. Anda harus memberikan satu nilai untuk setiap tag sesi. AWS STS tidak mendukung tag sesi multi-nilai.

Example AssumeRole CLIPermintaan contoh

```
aws sts assume-role \
```

```
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

Meneruskan tag sesi dengan menggunakan AssumeRoleWithSAML

`AssumeRoleWithSAML`Operasi mengotentikasi dengan federasi SAML berbasis. Operasi ini mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang penggunaan federasi SAML berbasis untuk AWS Management Console akses, lihatMengaktifkan pengguna federasi SAMP 2.0 untuk mengakses AWS Management Console. Untuk detail tentang AWS CLI atau AWS API akses, lihatSAML2.0 federasi. Untuk tutorial tentang mengkonfigurasi SAML federasi untuk pengguna Active Directory Anda, lihat AWS Otentikasi Federasi dengan Active Directory Federation Services (ADFS) di Blog Keamanan. AWS

Sebagai administrator, Anda dapat mengizinkan anggota direktori perusahaan Anda untuk bergabung AWS menggunakan AWS STS `AssumeRoleWithSAML` operasi. Untuk melakukannya, Anda harus menyelesaikan tugas berikut:

1. Konfigurasikan jaringan Anda sebagai SAML penyedia untuk AWS.
2. Buat SAML penyedia di IAM
3. Konfigurasikan peran dan izin AWS untuk pengguna federasi Anda
4. Selesai mengonfigurasi SAML IDP dan membuat pernyataan untuk respons otentikasi SAML

AWS termasuk penyedia identitas dengan end-to-end pengalaman bersertifikat untuk tag sesi dengan solusi identitas mereka. Untuk mempelajari cara menggunakan penyedia identitas ini untuk mengonfigurasi tag sesi, lihat Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS.

Untuk meneruskan SAML atribut sebagai tag sesi, sertakan `Attribute` elemen dengan `Name` atribut yang disetel kehttps://aws.amazon.com/SAML/Attributes/PrincipalTag:*{TagKey}*. Gunakan elemen `AttributeValue` untuk menentukan nilai tanda. Menyertakan elemen `Attribute` terpisah untuk setiap tag sesi.

Misalnya, anggap bahwa Anda ingin meneruskan atribut identitas berikut sebagai tag sesi:

- `Project:Automation`

- `CostCenter:12345`
- `Department:Engineering`

Untuk meneruskan atribut ini, sertakan elemen-elemen berikut dalam SAML pernyataan Anda.

Example Contoh cuplikan pernyataan SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

Untuk mengatur tag sebelumnya sebagai transitif, sertakan `Attribute` elemen lain dengan `Name` atribut yang disetel ke. `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys` Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Untuk mengatur `Project` dan `Department` tag sebagai transitif, gunakan atribut multi-nilai berikut:

Example Contoh cuplikan pernyataan SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

Meneruskan tag sesi dengan menggunakan AssumeRoleWithWebIdentity

Gunakan federasi yang sesuai dengan OpenID Connect (OIDC) untuk mengautentikasi operasi. `AssumeRoleWithWebIdentity` Operasi ini mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang penggunaan federasi identitas web untuk AWS Management Console akses, lihatOIDCfederasi.

Untuk meneruskan tag sesi dari OpenID Connect (OIDC), Anda harus menyertakan tag sesi di JSON Web Token (JWT) saat Anda mengirimkan permintaan. `AssumeRoleWithWebIdentity` Untuk

mempelajari lebih lanjut tentang OIDC token dan klaim, lihat Menggunakan Token dengan Kumpulan Pengguna di Panduan Amazon Cognito Pengembang.

AWS mendukung dua format klaim untuk menyertakan tag sesi diJWT:

- Format klaim bersarang
- Format klaim yang diratakan

Format klaim bersarang

Format klaim bersarang menggunakan struktur dalam `https://aws.amazon.com/tags` namespace di. JWT Dalam format ini:

- Tag utama direpresentasikan sebagai objek bersarang di bawah `principal_tags` kunci.
- Setiap tag utama dapat memiliki satu atau lebih nilai dalam array.
- Tombol tag transitif direpresentasikan dalam array di bawah `transitive_tag_keys` kunci.
- Keduanya `principal_tags` dan `transitive_tag_keys` bersarang di bawah `https://aws.amazon.com/tags` namespace.

Contoh berikut menunjukkan decoded JWT menggunakan format objek bersarang:

Example Contoh Token JSON Web yang diterjemahkan menggunakan format klaim bersarang

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
```

```
            "CostCenter"
        ]
    }
}
```

Format klaim yang diratakan

Format klaim yang diratakan kompatibel dengan penyedia identitas yang tidak mendukung objek bersarang dalam JWT klaim, seperti Microsoft Entra ID. Dalam format ini:

- Tag utama direpresentasikan sebagai klaim terpisah dengan awalan`https://aws.amazon.com/tags/principal_tags/`.
- Setiap tag utama adalah nilai string tunggal.
- Kunci tag transitif direpresentasikan dalam satu klaim sebagai array string dengan awalan. `https://aws.amazon.com/tags/transitive_tag_keys`

Sekarang, mari kita lihat bagaimana informasi yang sama direpresentasikan menggunakan format klaim yang diratakan:

Example Contoh Token JSON Web yang diterjemahkan menggunakan format klaim yang diratakan

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags/principal_tags/Project": "Automation",
    "https://aws.amazon.com/tags/principal_tags/CostCenter": "987654",
    "https://aws.amazon.com/tags/principal_tags/Department": "Engineering",
    "https://aws.amazon.com/tags/transitive_tag_keys": [
        "Project",
        "CostCenter"
    ]
}
```

Kedua JWT contoh yang diterjemahkan menunjukkan panggilan ke `AssumeRoleWithWebIdentity` dengan tag`Project`,`CostCenter`, dan `Department` sesi. Kedua token menetapkan `Project` dan

`CostCenter` tag sebagai transitif. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Format klaim yang diratakan mencapai hasil yang sama dengan format klaim bersarang tetapi menggunakan struktur pipih untuk tag. Ini memungkinkan Anda untuk menyertakan tag sesi di lingkungan di mana JSON objek bersarang tidak didukung dalam JWT klaim. Saat menggunakan salah satu format, pastikan bahwa penyedia identitas Anda dikonfigurasi untuk mengeluarkan token dengan struktur klaim yang sesuai. AWS mendukung kedua format klaim, sehingga Anda dapat memilih salah satu yang paling sesuai dengan persyaratan spesifik penyedia identitas Anda.

Meneruskan tag sesi dengan menggunakan GetFederationToken

`GetFederationToken`Ini memungkinkan Anda untuk menyatukan pengguna Anda. Operasi ini mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk menambahkan tag ke sesi pengguna federasi Anda, gunakan `--tags` AWS CLI opsi atau `Tags` AWS API parameter. Anda tidak dapat menyetel tag sesi sebagai transitif saat Anda menggunakan`GetFederationToken`, karena Anda tidak dapat menggunakan kredenal sementara untuk mengambil peran. Anda tidak dapat menggunakan rantai peran dalam kasus ini.

Contoh berikut menunjukkan permintaan sampel menggunakan `GetFederationToken`. Dalam contoh ini, saat meminta token, Anda membuat sesi bernama `my-fed-user`. Anda menambahkan pasangan nilai kunci tanda sesi `Project` = `Automation` dan `Department` = `Engineering`.

Example GetFederationToken CLIPermintaan contoh

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

Saat Anda menggunakan kredensi sementara yang dikembalikan oleh `GetFederationToken` operasi, tag utama sesi menyertakan tag pengguna dan tag sesi yang diteruskan.

Merangkai peran dengan tag sesi

Anda dapat mengasumsikan satu peran dan kemudian menggunakan kredensial sementara untuk mengasumsikan peran lain. Anda dapat melanjutkan dari sesi ke sesi. Ini disebut perangkaian peran. Saat Anda meneruskan tag sesi sembari mengasumsikan peran, Anda dapat mengatur kunci sebagai transitif. Hal ini memastikan bahwa tag sesi tersebut diteruskan ke sesi berikutnya dalam rantai peran. Anda tidak dapat mengatur tag peran sebagai transitif. Untuk meneruskan tag ini ke sesi berikutnya, tentukan tag tersebut sebagai tag sesi.

> **Note**
>
> Tag transitif bertahan selama rantai peran dan menggantikan `ResourceTag` nilai yang cocok setelah evaluasi kebijakan kepercayaan peran.

Contoh berikut menunjukkan bagaimana AWS STS meneruskan tag sesi, tag transitif, dan tag peran ke sesi berikutnya dalam rantai peran.

Dalam contoh skenario rantai peran ini, Anda menggunakan kunci akses IAM pengguna AWS CLI untuk mengambil peran bernama`Role1`. Anda kemudian menggunakan kredensial sesi yang dihasilkan untuk mengambil peran kedua bernama `Role2`. Anda kemudian dapat menggunakan kredensial sesi kedua untuk mengambil peran ketiga bernama `Role3`. Permintaan ini terjadi sebagai tiga operasi terpisah. Setiap peran sudah ditandai. IAM Dan dalam setiap permintaan, Anda meneruskan tag sesi tambahan.

Saat Anda merangkai peran, Anda dapat memastikan bahwa tag dari sesi sebelumnya tetap berlanjut ke sesi berikutnya. Untuk melakukan ini menggunakan `assume-role` CLI perintah, Anda harus meneruskan tag sebagai tag sesi dan mengatur tag sebagai transitif. Anda meneruskan tag `Star` = 1 sebagai tag sesi. Perintah juga melampirkan tag `Heart` = 1 ke peran dan berlaku sebagai tag utama saat Anda menggunakan sesi. Namun, Anda juga menginginkan tag `Heart` = 1 tag untuk diteruskan secara otomatis ke sesi kedua atau ketiga. Untuk melakukannya, Anda menyertakannya secara manual sebagai tag sesi. Tag utama sesi yang dihasilkan mencakup kedua tag ini, dan menetapkannya sebagai transitif.

Anda melakukan permintaan ini menggunakan AWS CLI perintah berikut:

Example AssumeRole CLIPermintaan contoh

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

Anda kemudian menggunakan kredensials untuk sesi tersebut guna mengambil `Role2`. Perintah melampirkan tag `Sun` = 2 ke peran kedua dan berlaku sebagai tag utama saat Anda menggunakan sesi kedua. `Star`Tag `Heart` dan mewarisi tag sesi transitif di sesi pertama. Sesi kedua yang dihasilkan tag utama adalah `Heart` =1, `Star` =1, dan `Sun` =2. `Heart`dan `Star` terus menjadi

transitif. SunTag yang Role2 dilampirkan tidak ditandai sebagai transitif karena bukan tag sesi. Sesi mendatang tidak mewarisi tag ini.

Anda melakukan permintaan kedua ini menggunakan AWS CLI perintah berikut:

Example AssumeRole CLIPermintaan contoh

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

Anda kemudian menggunakan kredensial kedua guna mengambil Role3. Tanda prinsipal untuk sesi ketiga berasal dari tanda sesi baru, tanda sesi transitif yang diturunkan, dan tanda peran. 1Tag Heart = 1 dan Star = pada sesi kedua diwarisi dari tag sesi transitif di sesi pertama. Jika Anda mencoba meneruskan tag 2 sesi Sun =, operasi gagal. Tag sesi Star = 1 yang diwarisi menggantikan tag role Star =. 3 Dalam rantai peran, nilai tag transitif mengesampingkan peran yang cocok dengan ResourceTag nilai setelah evaluasi kebijakan kepercayaan peran. Dalam contoh ini, jika Role3 menggunakan Star sebagai ResourceTag dalam kebijakan kepercayaan peran, dan menetapkan ResourceTag nilai ke nilai tag transitif dari sesi peran pemanggilan. LightningTag peran juga berlaku untuk sesi ketiga, dan tidak ditetapkan sebagai transitif.

Anda melakukan permintaan ketiga menggunakan AWS CLI perintah berikut:

Example AssumeRole CLIPermintaan contoh

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Menggunakan tag sesi untuk ABAC

Attribute-based access control (ABAC) menggunakan strategi otorisasi yang mendefinisikan izin berdasarkan atribut tag.

Jika perusahaan Anda menggunakan penyedia identitas SAML berbasis OIDC atau (iDP) untuk mengelola identitas pengguna, Anda dapat mengonfigurasi pernyataan Anda untuk meneruskan tag sesi. AWS Misalnya, dengan identitas pengguna perusahaan, ketika karyawan Anda bergabung AWS, AWS menerapkan atribut mereka pada prinsipal yang dihasilkan. Anda kemudian dapat menggunakan ABAC untuk mengizinkan atau menolak izin berdasarkan atribut tersebut. Untuk detailnya, lihat IAMtutorial: Gunakan tag SAML sesi untuk ABAC.

Untuk informasi selengkapnya tentang menggunakan Pusat IAM Identitas denganABAC, lihat Atribut untuk kontrol akses di Panduan AWS IAM Identity Center Pengguna.

Melihat tag sesi di CloudTrail

Anda dapat menggunakan AWS CloudTrail untuk melihat permintaan yang digunakan untuk mengambil peran atau pengguna federasi. File CloudTrail log mencakup informasi tentang tag utama untuk peran yang diasumsikan atau sesi pengguna federasi. Untuk informasi selengkapnya, lihat Penebangan IAM dan AWS STS APIpanggilan dengan AWS CloudTrail.

Misalnya, asumsikan bahwa Anda membuat AWS STS `AssumeRoleWithSAML` permintaan, meneruskan tag sesi, dan menetapkan tag tersebut sebagai transitif. Anda dapat menemukan informasi berikut di CloudTrail log Anda.

Example Contoh AssumeRoleWith SAML CloudTrail log

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

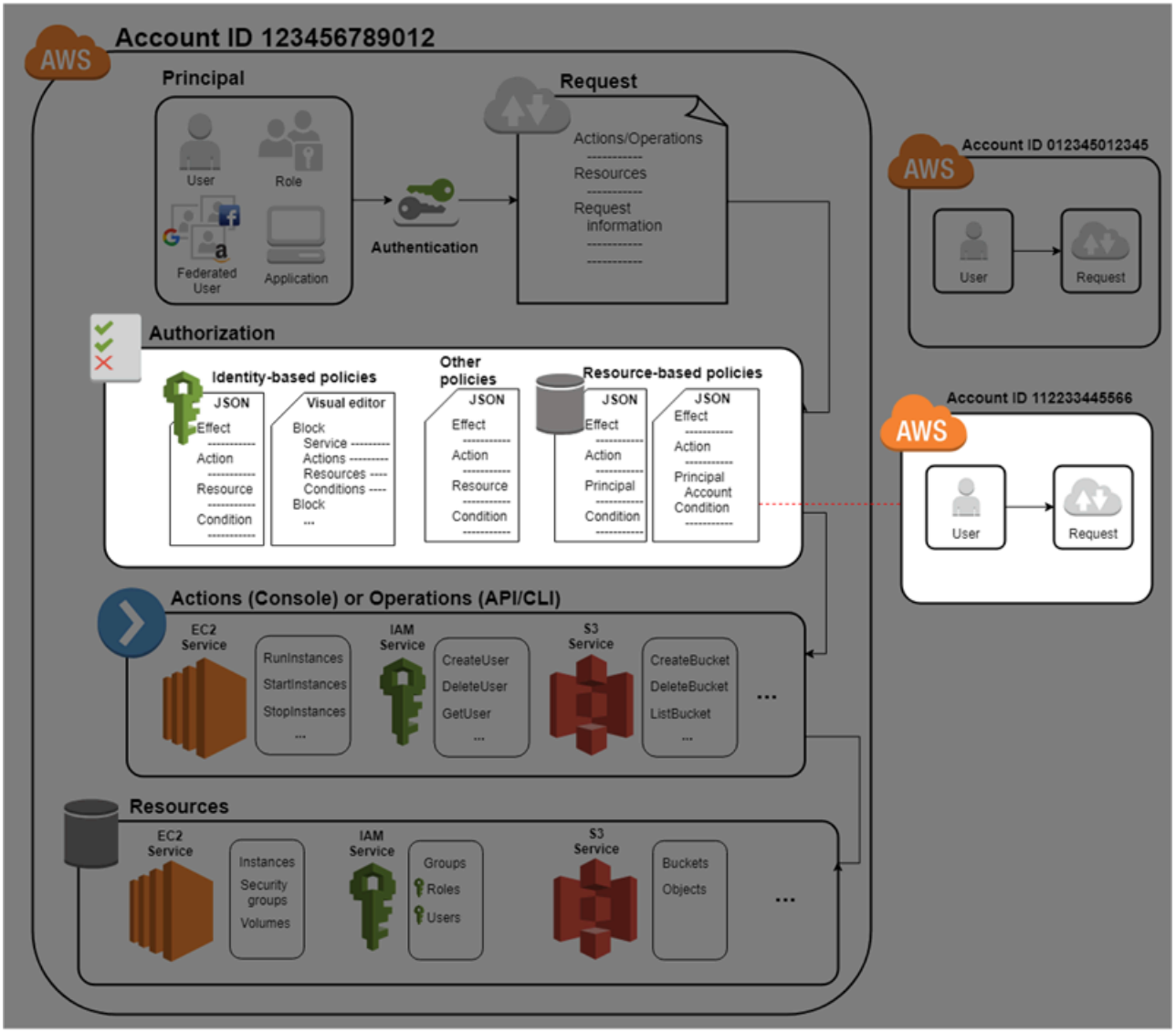
Anda dapat melihat contoh CloudTrail log berikut untuk melihat peristiwa yang menggunakan tag sesi.

- Contoh AWS STS APIperistiwa rantai peran dalam file CloudTrail log
- Contoh SAML AWS STS APIperistiwa dalam file CloudTrail log
- Contoh OIDC AWS STS APIperistiwa dalam file CloudTrail log

Manajemen akses untuk AWS sumber daya

AWS Identity and Access Management (IAM) adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya dengan aman. Ketika kepala sekolah membuat permintaan AWS, kode AWS penegakan memeriksa apakah prinsipal diautentikasi (masuk) dan diotorisasi (memiliki izin). Anda mengelola akses AWS dengan membuat kebijakan dan melampirkannya ke IAM identitas atau AWS sumber daya. Kebijakan adalah JSON dokumen AWS yang, ketika dilampirkan pada identitas atau sumber daya, menentukan izinnya. Untuk informasi selengkapnya tentang jenis dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

Untuk perincian tentang proses autentikasi dan otorisasi lainnya, lihat Cara kerja IAM.



Selama otorisasi, kode AWS penegakan menggunakan nilai dari konteks permintaan untuk memeriksa kebijakan yang cocok dan menentukan apakah akan mengizinkan atau menolak permintaan.

AWS memeriksa setiap kebijakan yang berlaku untuk konteks permintaan. Jika satu kebijakan menolak permintaan tersebut, AWS tolak seluruh permintaan dan berhenti mengevaluasi kebijakan. Ini disebut penolakan secara tegas. Karena permintaan ditolak secara default, IAM otorisasi permintaan Anda hanya jika setiap bagian dari permintaan Anda diizinkan oleh kebijakan yang berlaku. logika evaluasi untuk permintaan dalam satu akun mengikuti aturan berikut:

- Secara default, semua permintaan ditolak secara implisit. (Atau, secara default, Pengguna root akun AWS memiliki akses penuh.)
- Izin eksplisit dalam kebijakan berbasis identitas atau berbasis sumber daya akan membatalkan pengaturan default ini.
- Jika ada batas izin, OrganizationsSCP, atau kebijakan sesi, mungkin akan mengganti izin dengan penolakan implisit.
- Penolakan secara tegas dalam kebijakan apa pun akan mengesampingkan izin apa pun.

Setelah permintaan Anda diautentikasi dan diotorisasi, AWS setujui permintaan tersebut. Jika Anda perlu mengajukan permintaan di akun yang berbeda, kebijakan di akun lain harus memungkinkan Anda mengakses sumber daya. Selain itu, IAM entitas yang Anda gunakan untuk membuat permintaan harus memiliki kebijakan berbasis identitas yang memungkinkan permintaan tersebut.

Sumber daya manajemen akses

Untuk informasi selengkapnya tentang izin dan tentang pembuatan kebijakan, lihat sumber daya berikut:

Entri berikut di Blog AWS Keamanan mencakup cara-cara umum untuk menulis kebijakan untuk akses ke bucket dan objek Amazon S3.

- IAMKebijakan Penulisan: Cara Memberikan Akses ke Bucket Amazon S3
- IAMKebijakan penulisan: Berikan Akses ke Folder Khusus Pengguna di Bucket Amazon S3
- IAMKebijakan dan Kebijakan Bucket danACLs! Astaga! (Mengontrol Akses ke Sumber Daya S3)
- Primer tentang Izin RDS Tingkat Sumber Daya
- Mengungkap Izin Tingkat Sumber Daya EC2

Kebijakan dan izin di AWS Identity and Access Management

Kelola akses AWS dengan membuat kebijakan dan melampirkannya ke IAM identitas (pengguna, grup pengguna, atau peran) atau AWS sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika IAM prinsipal (pengguna atau peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai JSON dokumen. AWS mendukung enam jenis kebijakan: kebijakan berbasis identitas, kebijakan berbasis sumber daya, batas izin, kebijakan kontrol layanan Organisasi (), daftar kontrol akses (SCPs), dan kebijakan sesi. ACLs

IAMkebijakan menentukan izin untuk tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasi. Misalnya, jika kebijakan mengizinkan GetUsertindakan, maka pengguna dengan kebijakan itu bisa mendapatkan informasi pengguna dari AWS Management Console, AWS CLI, atau AWS API. Saat Anda membuat IAM pengguna, Anda dapat memilih untuk mengizinkan konsol atau akses terprogram. Jika akses konsol diizinkan, IAM pengguna dapat masuk ke konsol menggunakan kredenal masuknya. Jika akses terprogram diizinkan, pengguna dapat menggunakan kunci akses untuk bekerja dengan CLI atauAPI.

Jenis kebijakan

Jenis kebijakan berikut, yang tercantum dalam urutan dari yang paling sering digunakan hingga yang jarang digunakan, tersedia untuk digunakan di AWS. Untuk perincian selengkapnya, lihat bagian di bawah ini untuk setiap jenis kebijakan.

- Kebijakan berbasis identitas - Lampirkan kebijakan terkelola dan sebaris ke IAM identitas (pengguna, grup tempat pengguna berada, atau peran). Kebijakan berbasis identitas memberikan izin kepada sebuah identitas.
- Kebijakan berbasis sumber daya – Lampirkan kebijakan yang sesuai ke sumber daya. Contoh paling umum dari kebijakan berbasis sumber daya adalah kebijakan bucket Amazon S3 dan kebijakan kepercayaan peran. IAM Kebijakan berbasis sumber daya memberikan izin kepada prinsipal yang ditentukan dalam kebijakan. Prinsipal dapat berada di akun yang sama dengan sumber daya atau di akun lain.
- Batas izin — Gunakan kebijakan terkelola sebagai batas izin untuk IAM entitas (pengguna atau peran). Kebijakan tersebut menentukan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas, tetapi tidak memberikan izin. Batas izin tidak menentukan izin maksimum yang dapat diberikan oleh kebijakan berbasis sumber daya kepada entitas.

- Organizations SCPs — Gunakan kebijakan kontrol AWS Organizations layanan (SCP) untuk menentukan izin maksimum bagi anggota akun organisasi atau unit organisasi (OU). SCPsmembatasi izin yang diberikan oleh kebijakan berbasis identitas atau kebijakan berbasis sumber daya kepada entitas (pengguna atau peran) dalam akun, tetapi tidak memberikan izin.
- Daftar kontrol akses (ACLs) — Gunakan ACLs untuk mengontrol prinsipal mana di akun lain yang dapat mengakses sumber daya yang dilampirkan. ACL ACLsmirip dengan kebijakan berbasis sumber daya, meskipun mereka adalah satu-satunya jenis kebijakan yang tidak menggunakan struktur dokumen kebijakan. JSON ACLsadalah kebijakan izin lintas akun yang memberikan izin kepada prinsipal yang ditentukan. ACLstidak dapat memberikan izin kepada entitas dalam akun yang sama.
- Kebijakan sesi — Lulus kebijakan sesi lanjutan saat Anda menggunakan AWS CLI atau AWS API untuk mengambil peran atau pengguna gabungan. Kebijakan sesi membatasi izin yang diberikan oleh peran atau kebijakan berbasis identitas pengguna ke sesi. Kebijakan sesi membatasi izin untuk sesi yang dibuat, tetapi tidak memberikan izin. Untuk informasi lebih lanjut, lihat Kebijakan Sesi

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan JSON izin yang mengontrol tindakan apa yang dapat dilakukan identitas (pengguna, grup pengguna, dan peran), pada sumber daya mana, dan dalam kondisi apa. Kebijakan berbasis identitas selanjutnya dapat dikategorikan menjadi:

- Kebijakan terkelola — Kebijakan berbasis identitas mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Anda. Akun AWS Ada dua jenis kebijakan terkelola:
  - AWS kebijakan terkelola — Kebijakan terkelola yang dibuat dan dikelola oleh AWS.
  - Kebijakan terkelola pelanggan — Kebijakan terkelola yang Anda buat dan kelola di Anda Akun AWS. Kebijakan yang dikelola pelanggan memberikan kontrol yang lebih tepat atas kebijakan Anda daripada kebijakan yang AWS dikelola.
- Kebijakan inline – Kebijakan yang Anda tambahkan secara langsung ke satu pengguna, grup, atau peran. Kebijakan inline mempertahankan one-to-one hubungan yang ketat antara kebijakan dan identitas. Mereka dihapus ketika Anda menghapus identitas.

Untuk mempelajari cara memilih antara kebijakan terkelola dan kebijakan inline, lihat Pilih antara kebijakan terkelola dan kebijakan inline.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen JSON kebijakan yang Anda lampirkan ke sumber daya seperti bucket Amazon S3. Kebijakan ini memberikan izin prinsipal yang ditentukan untuk melakukan tindakan tertentu pada sumber daya tersebut dan menetapkan di bawah ketentuan yang berlaku. Kebijakan berbasis sumber daya merupakan kebijakan inline. Tidak ada kebijakan berbasis sumber daya terkelola.

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan seluruh akun atau IAM entitas di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya terpisah Akun AWS, Anda juga harus menggunakan kebijakan berbasis identitas untuk memberikan akses utama ke sumber daya. Namun, jika kebijakan berbasis sumber daya memberikan akses ke prinsipal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk petunjuk langkah demi langkah untuk memberikan akses lintaslayanan, lihat IAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran.

IAMLayanan ini hanya mendukung satu jenis kebijakan berbasis sumber daya yang disebut kebijakan kepercayaan peran, yang melekat pada peran. IAM IAMPeran adalah identitas dan sumber daya yang mendukung kebijakan berbasis sumber daya. Untuk alasan itu, Anda harus melampirkan kebijakan kepercayaan dan kebijakan berbasis identitas ke suatu peran. IAM Kebijakan kepercayaan menentukan entitas prinsipal mana (akun, pengguna, peran, dan pengguna gabungan) yang dapat memegang peran tersebut. Untuk mempelajari perbedaan IAM peran dari kebijakan berbasis sumber daya lainnya, lihat. Akses sumber daya lintas akun di IAM

Untuk melihat mana layanan lainnya yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari lebih lanjut tentang kebijakan berbasis sumber daya, lihat Kebijakan berbasis identitas dan kebijakan berbasis sumber daya. Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? .

Batas izin IAM

Batas izin adalah fitur lanjutan tempat Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas. IAM Saat Anda menetapkan batas izin untuk suatu entitas, entitas hanya dapat melakukan tindakan yang diizinkan oleh kedua kebijakan berbasis identitas dan batas izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran sebagai prinsipal tidak dibatasi oleh batas izin. Namun, jika kebijakan berbasis sumber daya menentukan peran sebagai prinsipal, maka kebijakan tersebut dibatasi oleh batas izin. Penolakan

eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi lebih lanjut tentang batas izin, lihat Batas izin untuk entitas IAM.

Kebijakan kontrol layanan (SCPs)

AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCPsadalah JSON kebijakan yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU). SCPMembatasi izin untuk entitas di akun anggota, termasuk masing-masing Pengguna root akun AWS. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin.

Untuk informasi selengkapnya tentang Organizations danSCPs, lihat Kebijakan kontrol layanan di Panduan AWS Organizations Pengguna.

Daftar kontrol akses (ACLs)

Access control lists (ACLs) adalah kebijakan layanan yang memungkinkan Anda mengontrol prinsipal mana di akun lain yang dapat mengakses sumber daya. ACLstidak dapat digunakan untuk mengontrol akses untuk prinsipal dalam akun yang sama. ACLsmirip dengan kebijakan berbasis sumber daya, meskipun mereka adalah satu-satunya jenis kebijakan yang tidak menggunakan format dokumen kebijakan. JSON Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnyaACLs, lihat Ringkasan Daftar Kontrol Akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

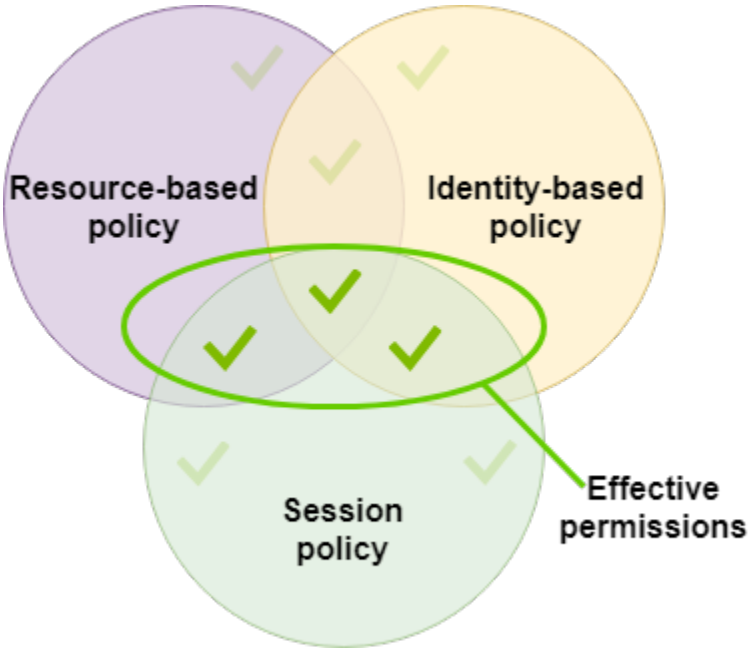
Kebijakan sesi

Kebijakan sesi adalah kebijakan lanjutan yang Anda teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. Izin untuk sesi adalah persimpangan kebijakan berbasis identitas untuk IAM entitas (pengguna atau peran) yang digunakan untuk membuat sesi dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.

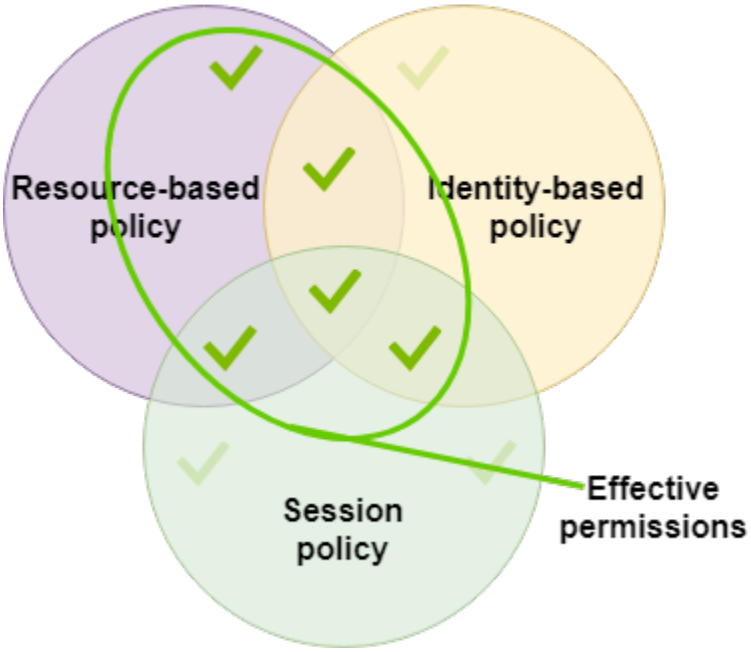
Anda dapat membuat sesi peran dan meneruskan kebijakan sesi secara terprogram menggunakan`AssumeRole`,`AssumeRoleWithSAML`, atau `AssumeRoleWithWebIdentity` API operasi. Anda dapat meneruskan satu dokumen kebijakan sesi JSON inline menggunakan `Policy` parameter. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Untuk informasi selengkapnya tentang cara membuat sesi peran, lihat Izin untuk kredensial keamanan sementara.

Saat Anda membuat sesi pengguna federasi, Anda menggunakan kunci akses IAM pengguna untuk memanggil operasi secara terprogram. `GetFederationToken` API Anda juga harus memberikan kebijakan sesi. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas dan kebijakan sesi. Untuk informasi selengkapnya tentang cara membuat sesi pengguna gabungan, lihat Meminta kredensi melalui pialang identitas khusus.

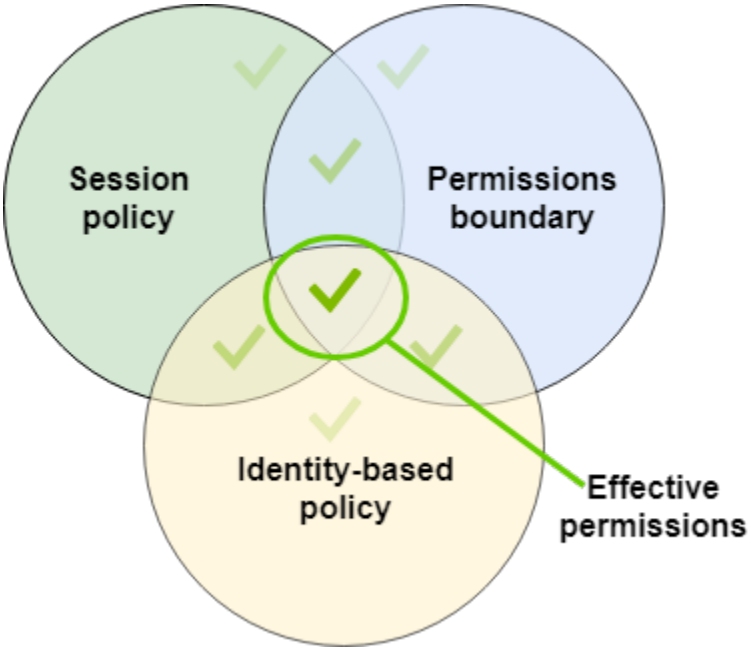
Kebijakan berbasis sumber daya dapat menentukan pengguna atau peran sebagai prinsipal. ARN Dalam hal ini, izin dari kebijakan berbasis sumber daya ditambahkan ke peran atau kebijakan berbasis identitas pengguna sebelum sesi dibuat. Kebijakan sesi membatasi total izin yang diberikan oleh kebijakan berbasis sumber daya dan kebijakan berbasis identitas. Izin sesi yang dihasilkan adalah persimpangan kebijakan sesi dan kebijakan berbasis sumber daya ditambah persimpangan kebijakan sesi dan kebijakan berbasis identitas.



Kebijakan berbasis sumber daya dapat menentukan sesi sebagai ARN prinsipal. Dalam hal ini, izin dari kebijakan berbasis sumber daya ditambahkan setelah sesi dibuat. Izin kebijakan berbasis sumber daya tidak terbatas oleh kebijakan sesi. Izin sesi yang dihasilkan mempunyai semua izin dari kebijakan berbasis sumber daya ditambah titik pertemuan antara kebijakan berbasis identitas dan kebijakan sesi.



Batas izin dapat mengatur izin maksimum untuk pengguna atau peran yang digunakan untuk membuat sebuah sesi. Dalam kasus ini, Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan kebijakan sesi, batas izin, dan kebijakan berbasis identitas. Namun, batas izin tidak membatasi izin yang diberikan oleh kebijakan berbasis sumber daya yang menentukan sesi yang dihasilkan. ARN



Kebijakan dan pengguna root

Pengguna root akun AWS Ini dipengaruhi oleh beberapa jenis kebijakan tetapi tidak yang lain. Anda tidak dapat melampirkan kebijakan berbasis identitas ke pengguna root, dan Anda tidak dapat menetapkan batas izin bagi pengguna root. Namun, Anda dapat menentukan pengguna root sebagai prinsipal dalam kebijakan berbasis sumber daya atau file. ACL Pengguna root masih menjadi anggota akun. Jika akun itu adalah anggota organisasi di AWS Organizations, pengguna root dipengaruhi oleh akun apa punSCPs.

Ikhtisar JSON kebijakan

Sebagian besar kebijakan disimpan AWS sebagai JSON dokumen. Kebijakan dan kebijakan berbasis identitas yang digunakan untuk menetapkan batas izin adalah dokumen JSON kebijakan yang Anda lampirkan ke pengguna atau peran. Kebijakan berbasis sumber daya adalah dokumen JSON kebijakan yang Anda lampirkan ke sumber daya. SCPsadalah dokumen JSON kebijakan dengan sintaks terbatas yang Anda lampirkan ke unit AWS Organizations organisasi (OU). ACLsjuga melekat pada sumber daya, tetapi Anda harus menggunakan sintaks yang berbeda. Kebijakan sesi adalah JSON kebijakan yang Anda berikan saat Anda mengambil peran atau sesi pengguna gabungan.

Anda tidak perlu memahami JSON sintaksnya. Anda dapat menggunakan editor visual AWS Management Console untuk membuat dan mengedit kebijakan yang dikelola pelanggan tanpa pernah menggunakanJSON. Namun, jika Anda menggunakan kebijakan sebaris untuk grup atau kebijakan kompleks, Anda tetap harus membuat dan mengedit kebijakan tersebut di JSON editor menggunakan konsol. Untuk informasi lebih lanjut tentang cara menggunakan editor visual, lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan dan Edit IAM kebijakan.

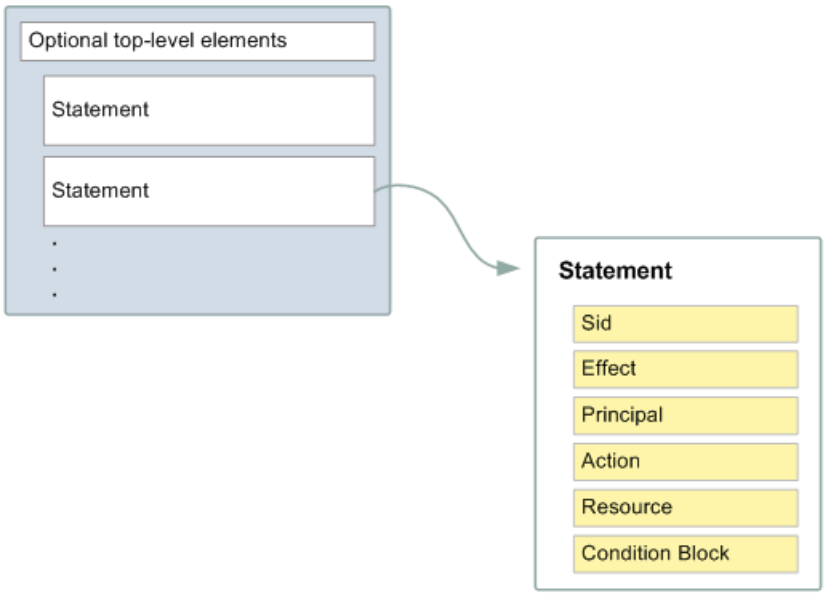
Saat Anda membuat atau mengedit JSON kebijakan, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAMmengidentifikasi kesalahan JSON sintaks, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat Validasi kebijakan IAMAccess Analyzer.

JSONstruktur dokumen kebijakan

Seperti yang diilustrasikan pada gambar berikut, dokumen JSON kebijakan mencakup elemen-elemen ini:

- Informasi opsional untuk seluruh kebijakan di bagian atas dokumen
- Satu atau lebih pernyataan individu

Setiap pernyataan mencakup informasi tentang satu izin. Jika kebijakan mencakup beberapa pernyataan, AWS terapkan logis `OR` di seluruh pernyataan saat mengevaluasinya. Jika beberapa kebijakan berlaku untuk permintaan, AWS terapkan logika `OR` di semua kebijakan tersebut saat mengevaluasinya.



Informasi dalam pernyataan terkandung di dalam rangkaian elemen.

- Version – Tentukan versi bahasa kebijakan yang ingin Anda gunakan. Kami menyarankan Anda menggunakan `2012-10-17` versi terbaru. Untuk informasi selengkapnya, silakan lihat IAMJSONelemen kebijakan: Version
- Statement – Gunakan elemen kebijakan utama ini sebagai wadah untuk elemen berikut. Anda dapat menyertakan lebih dari satu pernyataan dalam kebijakan.
- Sid (Opsional) – Sertakan ID pernyataan opsional untuk membedakan antara pernyataan Anda.
- Effect – Gunakan `Allow` atau `Deny` untuk menunjukkan apakah kebijakan mengizinkan atau menolak akses.
- Principal (Diwajibkan hanya dalam beberapa situasi) – Jika Anda membuat kebijakan berbasis sumber daya, Anda harus mengindikasikan akun, pengguna, peran, atau pengguna gabungan yang ingin Anda izinkan atau tolak aksesnya. Jika Anda membuat kebijakan IAM izin untuk

dilampirkan ke pengguna atau peran, Anda tidak dapat menyertakan elemen ini. Prinsipal tersirat sebagai pengguna atau peran tersebut.

- Action – Sertakan daftar tindakan yang diperbolehkan atau ditolak oleh kebijakan.
- Resource(Diperlukan hanya dalam beberapa keadaan) — Jika Anda membuat kebijakan IAM izin, Anda harus menentukan daftar sumber daya yang diterapkan tindakan tersebut. Jika Anda membuat kebijakan berbasis sumber daya, elemen ini bersifat opsional. Jika Anda tidak menyertakan elemen ini, maka sumber daya yang berlaku adalah sumber daya yang terkait dengan kebijakan tersebut.
- Condition (Opsional) – Tentukan keadaan di mana kebijakan memberikan izin.

Untuk mempelajari tentang ini dan elemen kebijakan yang lebih maju lainnya, lihat IAMJSONreferensi elemen kebijakan.

Beberapa pernyataan dan beberapa kebijakan

Jika Anda ingin menetapkan lebih dari satu izin untuk suatu entitas (pengguna atau peran), Anda dapat menggunakan beberapa pernyataan dalam satu kebijakan. Anda juga dapat melampirkan beberapa kebijakan. Jika Anda mencoba untuk menentukan beberapa izin dalam satu pernyataan, kebijakan Anda mungkin tidak memberikan akses yang Anda harapkan. Kami menyarankan Anda memecah kebijakan berdasarkan jenis sumber daya.

Karena ukuran kebijakan yang terbatas, mungkin perlu menggunakan beberapa kebijakan untuk izin yang lebih kompleks. Ini juga merupakan ide bagus untuk membuat pengelompokan fungsional izin dalam kebijakan pengelolaan pelanggan yang terpisah. Misalnya, Buat satu kebijakan untuk manajemen IAM pengguna, satu untuk manajemen mandiri, dan kebijakan lain untuk manajemen bucket S3. Terlepas dari kombinasi beberapa pernyataan dan beberapa kebijakan, AWS evaluasi kebijakan Anda dengan cara yang sama.

Misalnya, kebijakan berikut memiliki tiga pernyataan, yang masing-masing mendefinisikan satu set izin secara terpisah dalam satu akun. Pernyataan tersebut menjelaskan hal berikut:

- Pernyataan pertama, dengan sebuah `Sid` (ID Pernyataan) dari `FirstStatement`, memungkinkan pengguna dengan kebijakan terlampir untuk mengubah kata sandi milik mereka. Elemen `Resource` dalam pernyataan ini adalah “*“ (yang berarti “semua sumber daya”). Namun dalam praktiknya, `ChangePassword` API operasi (atau `change-password` CLI perintah yang setara) hanya memengaruhi kata sandi untuk pengguna yang membuat permintaan.

- Pernyataan kedua memungkinkan pengguna mencantumkan semua ember Amazon S3 di dalamnya. Akun AWS Elemen `Resource` dalam pernyataan ini adalah `"*"` (yang berarti “semua sumber daya”). Tetapi karena kebijakan tidak memberikan akses ke sumber daya di akun lain, pengguna hanya dapat mencantumkan bucket di akun mereka sendiri Akun AWS.
- Pernyataan ketiga memungkinkan pengguna membuat daftar dan mengambil objek apa pun yang ada di bucket bernama`amzn-s3-demo-bucket-confidential-data`, tetapi hanya jika pengguna diautentikasi dengan otentikasi multi-faktor (). MFA `Condition`Elemen dalam kebijakan memberlakukan MFA otentikasi.

  Saat pernyataan kebijakan memuat elemen `Condition`, pernyataan hanya berlaku saat elemen `Condition` dievaluasi menjadi benar. Dalam hal ini, `Condition` mengevaluasi ke true ketika pengguna MFA -otentikasi. Jika pengguna tidak MFA diautentikasi, ini `Condition` dievaluasi menjadi false. Dalam hal ini, pernyataan ketiga dalam kebijakan ini tidak berlaku dan pengguna tidak memiliki akses ke bucket `amzn-s3-demo-bucket-confidential-data`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data",
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data/*"
```

```
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Contoh sintaks JSON kebijakan

Kebijakan berbasis identitas berikut mengizinkan prinsipal tersirat untuk mencantumkan satu bucket Amazon S3 dengan nama `amzn-s3-demo-bucket`:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
  }
}
```

Kebijakan berbasis sumber daya berikut dapat dilampirkan ke bucket Amazon S3 Kebijakan ini memungkinkan anggota tertentu Akun AWS untuk melakukan tindakan Amazon S3 apa pun dalam bucket yang diberi nama. `amzn-s3-demo-bucket` Hal ini mengizinkan tindakan apa pun yang dapat dilakukan pada bucket atau objek di dalamnya. (Karena kebijakan tersebut hanya memberikan kepercayaan ke akun, pengguna individu dalam akun tersebut harus tetap diberikan izin untuk tindakan Amazon S3 yang ditentukan.)

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }]
}
```

Untuk melihat contoh kebijakan untuk skenario umum, lihat Contoh kebijakan berbasis identitas IAM.

Berikan hak akses paling rendah

Saat Anda membuat IAM kebijakan, ikuti saran keamanan standar untuk memberikan hak istimewa paling sedikit, atau hanya memberikan izin yang diperlukan untuk melakukan tugas. Tentukan apa yang perlu dilakukan pengguna dan peran dan kemudian buat kebijakan yang memungkinkan mereka hanya melakukan tugas-tugas tersebut.

Mulai dengan satu set izin minimum dan berikan izin tambahan sesuai kebutuhan. Melakukan hal tersebut lebih aman daripada memulai dengan izin yang terlalu fleksibel, lalu mencoba memperketatnya nanti.

Sebagai alternatif dari hak istimewa terkecil, Anda dapat menggunakan kebijakan atau kebijakan AWS terkelola dengan * izin wildcard untuk memulai kebijakan. Pertimbangkan risiko keamanan memberikan izin kepada kepala sekolah Anda lebih banyak daripada yang mereka butuhkan untuk melakukan pekerjaan mereka. Pantau prinsipal tersebut untuk mempelajari izin mana yang mereka gunakan. Kemudian tulis kebijakan hak istimewa paling sedikit.

IAMmenyediakan beberapa opsi untuk membantu Anda memperbaiki izin yang Anda berikan.

- Memahami pengelompokan tingkat akses — Anda dapat menggunakan pengelompokan tingkat akses untuk memahami tingkat akses yang diberikan kebijakan. Tindakan kebijakan diklasifikasikan sebagai `List`, `Read`, `Write`, `Permissions management`, atau `Tagging`. Misalnya, Anda dapat memilih tindakan dari tingkat akses `List` dan `Read` untuk memberikan akses hanya-baca kepada pengguna Anda. Untuk mempelajari cara menggunakan ringkasan kebijakan untuk memahami izin tingkat akses, lihat Tingkat akses dalam ringkasan kebijakan.
- Validasi kebijakan Anda — Anda dapat melakukan validasi kebijakan menggunakan IAM Access Analyzer saat membuat dan mengedit kebijakan. JSON Sebaiknya Anda meninjau dan memvalidasi semua kebijakan yang ada. IAMAccess Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan untuk memvalidasi kebijakan Anda. Ini menghasilkan peringatan keamanan ketika pernyataan dalam kebijakan Anda memungkinkan akses yang kita anggap terlalu permisif. Anda dapat menggunakan rekomendasi yang dapat ditindaklanjuti yang disediakan melalui peringatan keamanan saat Anda berupaya memberikan hak istimewa paling rendah. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan yang disediakan oleh IAM Access Analyzer, lihat Validasi kebijakan IAM Access Analyzer.
- Buat kebijakan berdasarkan aktivitas akses — Untuk membantu Anda menyaring izin yang Anda berikan, Anda dapat membuat IAM kebijakan yang didasarkan pada aktivitas akses untuk IAM

entitas (pengguna atau peran). IAMAccess Analyzer meninjau AWS CloudTrail log Anda dan menghasilkan templat kebijakan yang berisi izin yang telah digunakan oleh entitas dalam jangka waktu yang ditentukan. Anda dapat menggunakan templat untuk membuat kebijakan terkelola dengan izin berbutir halus, lalu melampirkannya ke entitas. IAM Dengan begitu, Anda hanya memberikan izin yang dibutuhkan pengguna atau peran untuk berinteraksi dengan AWS sumber daya untuk kasus penggunaan spesifik Anda. Untuk mempelajari selengkapnya, lihat Pembuatan kebijakan IAM Access Analyzer.

- Gunakan informasi yang diakses terakhir - Fitur lain yang dapat membantu dengan hak istimewa paling sedikit adalah informasi yang diakses terakhir. Lihat informasi ini di tab Access Advisor di halaman detail IAM konsol untuk IAM pengguna, grup, peran, atau kebijakan. Informasi yang diakses terakhir juga mencakup informasi tentang tindakan yang terakhir diakses untuk beberapa layanan, seperti Amazon,EC2, LambdaIAM, dan Amazon S3. Jika Anda masuk menggunakan kredensi akun AWS Organizations manajemen, Anda dapat melihat informasi layanan yang terakhir diakses di AWS Organizationsbagian IAM konsol. Anda juga dapat menggunakan AWS CLI atau AWS API untuk mengambil laporan untuk informasi yang terakhir diakses untuk entitas atau kebijakan di IAM atau Organizations. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu sehingga Anda dapat menyempurnakan kebijakan Anda atau IAM Organizations agar lebih mematuhi prinsip hak istimewa yang paling rendah. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.
- Tinjau peristiwa akun di AWS CloudTrail — Untuk mengurangi izin lebih lanjut, Anda dapat melihat peristiwa akun Anda di Riwayat AWS CloudTrail acara. CloudTrail log peristiwa mencakup informasi peristiwa terperinci yang dapat Anda gunakan untuk mengurangi izin kebijakan. Log hanya mencakup tindakan dan sumber daya yang dibutuhkan IAM entitas Anda. Untuk informasi selengkapnya, lihat Melihat CloudTrail Acara di CloudTrail Konsol di Panduan AWS CloudTrail Pengguna.

Untuk informasi selengkapnya, lihat topik kebijakan berikut untuk layanan individual, yang memberikan contoh cara menulis kebijakan untuk sumber daya khusus layanan.

- Menggunakan kebijakan berbasis sumber daya untuk DynamoDB di Panduan Pengembang Amazon DynamoDB
- Kebijakan bucket untuk Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon
- Ikhtisar Daftar Kontrol Akses (ACL) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon

Kebijakan terkelola dan kebijakan inline

Saat menetapkan izin untuk identitasIAM, Anda harus memutuskan apakah akan menggunakan kebijakan AWS terkelola, kebijakan yang dikelola pelanggan, atau kebijakan sebaris. Topik berikut memberikan informasi lebih lanjut tentang masing-masing jenis kebijakan berbasis identitas dan kapan menggunakannya.

Topik

AWS kebijakan terkelola

Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Kebijakan mandiri berarti bahwa kebijakan tersebut memiliki Nama Sumber Daya Amazon (ARN) sendiri yang menyertakan nama kebijakan. Misalnya, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` adalah kebijakan yang AWS dikelola. Untuk informasi lebih lanjut tentangARNs, lihatIAM ARNs. Untuk mengetahui daftar kebijakan AWS terkelola Layanan AWS, lihat kebijakan AWS terkelola.

AWS Kebijakan terkelola memudahkan Anda untuk menetapkan izin yang sesuai kepada pengguna, IAM grup, dan peran. Ini lebih cepat daripada menulis kebijakan sendiri, dan menyertakan izin untuk banyak kasus penggunaan umum.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. AWS terkadang memperbarui izin yang ditentukan dalam kebijakan AWS terkelola. AWS Kapan melakukannya, pembaruan memengaruhi semua entitas utama (IAMpengguna, IAM grup, dan IAM peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat AWS layanan baru diluncurkan atau API panggilan baru tersedia untuk layanan yang ada. Misalnya, kebijakan AWS terkelola yang disebut ReadOnlyAccessmenyediakan akses hanya-baca ke semua Layanan AWS dan sumber daya. Saat AWS meluncurkan layanan baru, AWS memperbarui ReadOnlyAccesskebijakan untuk menambahkan izin hanya-baca untuk layanan baru. Izin yang diperbarui diterapkan pada semua entitas prinsipal yang kebijakannya dilampirkan.

Kebijakan AWS terkelola akses penuh: Ini menentukan izin untuk administrator layanan dengan memberikan akses penuh ke layanan. Contohnya termasuk:

- AmazonDynamoDBFullAccess
- IAMFullAccess

Kebijakan yang AWS dikelola pengguna daya: Ini menyediakan akses penuh ke Layanan AWS dan sumber daya, tetapi tidak memungkinkan mengelola pengguna dan IAM grup. Contohnya termasuk:

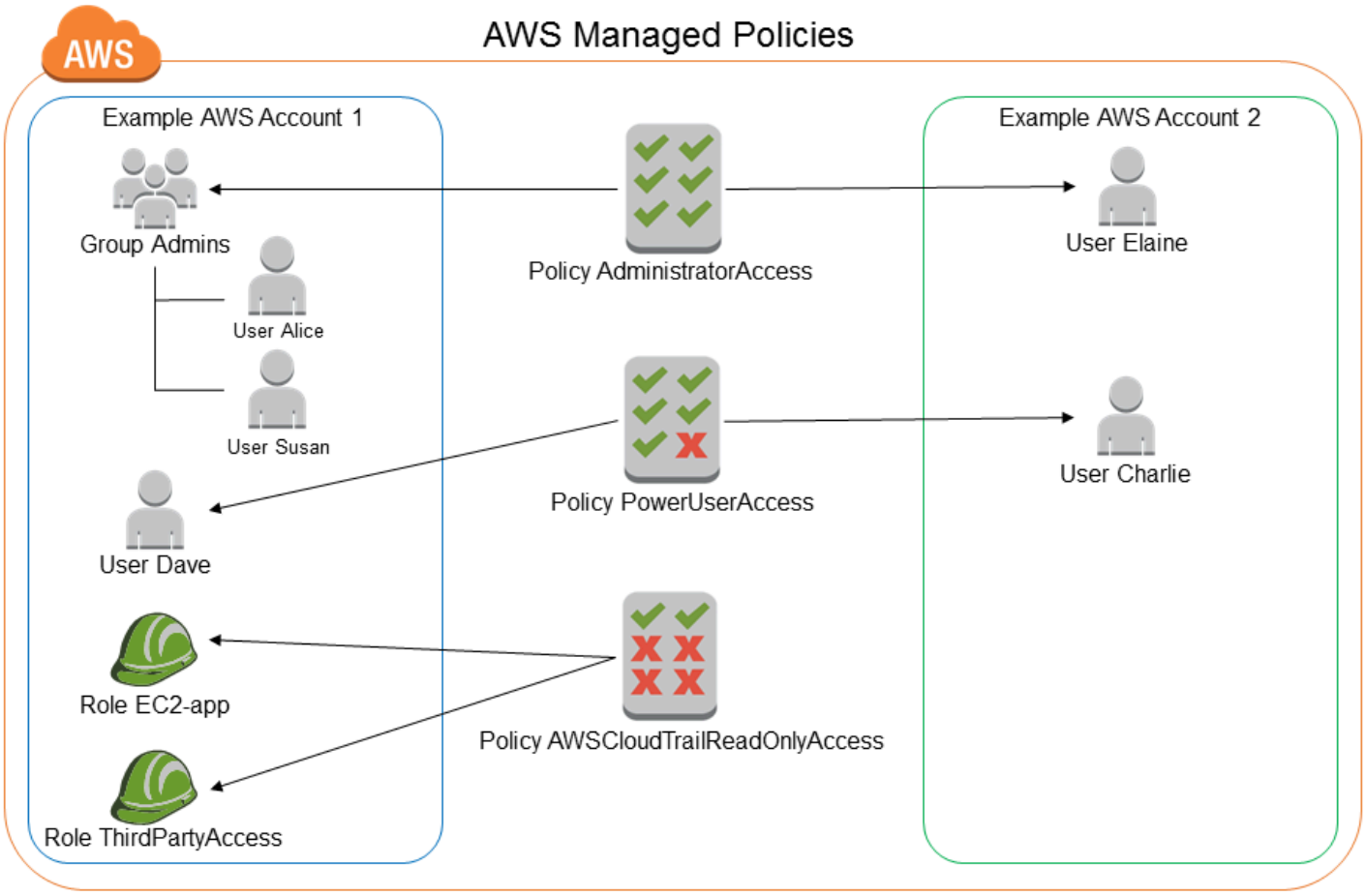
- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

Kebijakan AWS terkelola akses sebagian: Kebijakan ini memberikan tingkat akses tertentu Layanan AWS tanpa mengizinkan izin tingkat akses manajemen izin. Contohnya termasuk:

- AmazonMobileAnalyticsWriteOnlyAccess
- Amazon EC2ReadOnlyAccess

Kebijakan yang AWS dikelola fungsi Job: Kebijakan ini selaras dengan fungsi pekerjaan yang umum digunakan di industri TI dan memfasilitasi pemberian izin untuk fungsi pekerjaan ini. Salah satu keuntungan utama menggunakan kebijakan fungsi pekerjaan adalah bahwa mereka dipertahankan dan diperbarui oleh AWS ketika layanan dan API operasi baru diperkenalkan. Misalnya, fungsi AdministratorAccesspekerjaan menyediakan akses penuh dan delegasi izin ke setiap layanan dan sumber daya di. AWS Kami menyarankan Anda menggunakan kebijakan ini hanya untuk administrator akun. Untuk pengguna listrik yang memerlukan akses penuh ke setiap layanan kecuali akses terbatas ke IAM dan Organizations, gunakan fungsi PowerUserAccesspekerjaan. Untuk daftar dan deskripsi kebijakan fungsi pekerjaan, lihat AWS kebijakan terkelola untuk fungsi pekerjaan.

Diagram berikut menggambarkan kebijakan yang AWS dikelola. Diagram menunjukkan tiga kebijakan AWS terkelola: AdministratorAccessPowerUserAccess,, dan AWS CloudTrail_ ReadOnlyAccess. Perhatikan bahwa kebijakan AWS terkelola tunggal dapat dilampirkan ke entitas utama yang berbeda Akun AWS, dan ke entitas utama yang berbeda dalam satu Akun AWS.



Kebijakan yang dikelola pelanggan

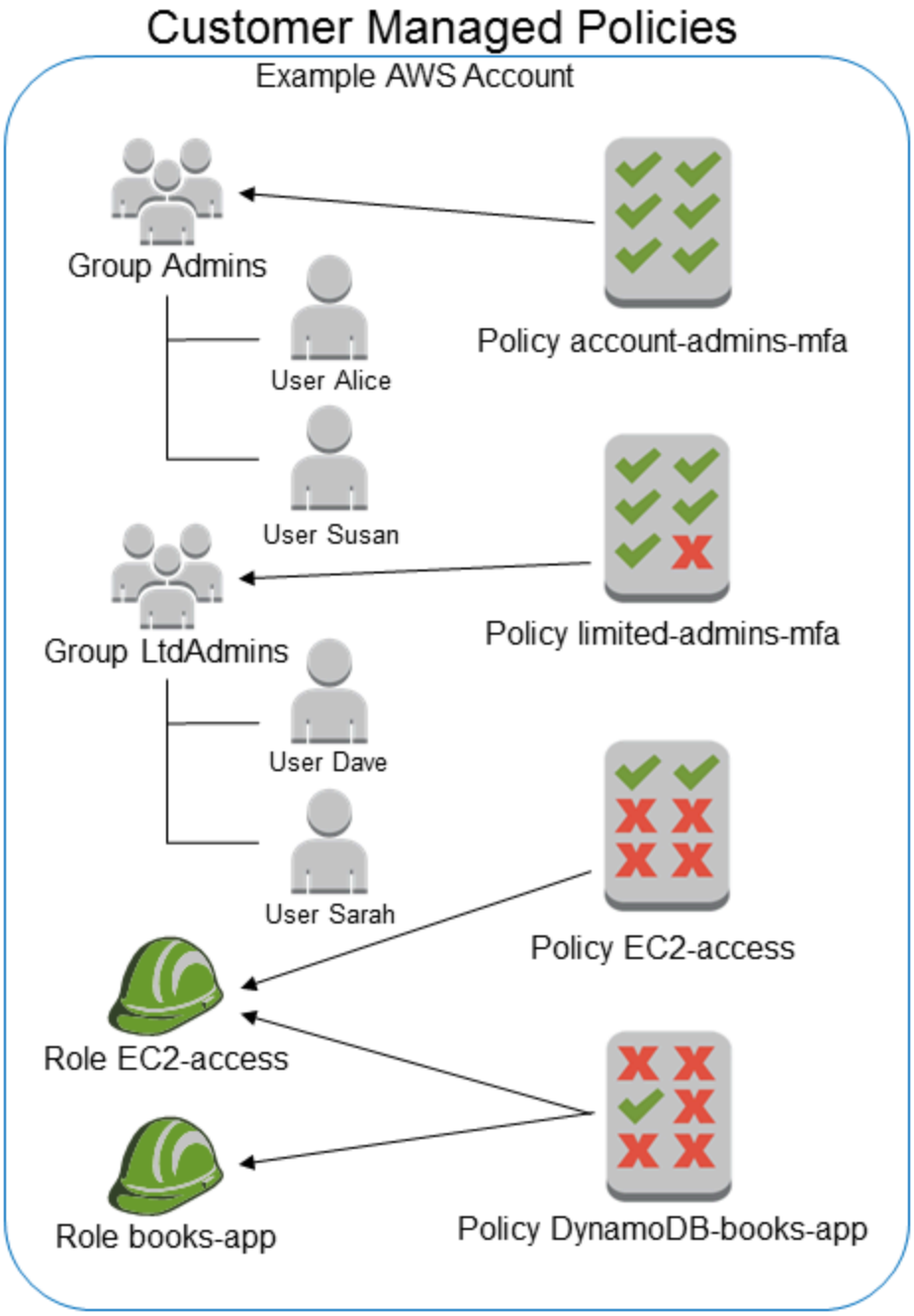
Anda dapat membuat kebijakan mandiri sendiri Akun AWS yang dapat Anda lampirkan ke entitas utama (IAMpengguna, IAM grup, dan IAM peran). Anda membuat kebijakan yang dikelola pelanggan ini untuk kasus penggunaan spesifik Anda, dan Anda dapat mengubah dan memperbaruinya sesering yang Anda suka. Seperti kebijakan AWS terkelola, saat Anda melampirkan kebijakan ke entitas utama, Anda memberi entitas izin yang ditentukan dalam kebijakan tersebut. Saat Anda memperbarui izin dalam kebijakan, perubahan akan diterapkan ke semua entitas utama yang dilampirkan kebijakan tersebut.

Cara terbaik untuk membuat kebijakan yang dikelola pelanggan adalah memulai dengan menyalin kebijakan terkelola AWS yang sudah ada. Dengan demikian Anda tahu bahwa kebijakan tersebut benar sejak awal dan yang perlu Anda lakukan hanyalah menyesuaikan itu dengan lingkungan Anda.

Diagram berikut menggambarkan kebijakan yang dikelola pelanggan. Setiap kebijakan adalah entitas IAM dengan Amazon Resource Name (ARN) miliknya sendiri yang menyertakan nama kebijakan.

Perhatikan bahwa kebijakan yang sama dapat dilampirkan ke beberapa entitas utama — misalnya, kebijakan DynamoDB-Books-App yang sama dilampirkan ke dua peran yang berbeda. IAM

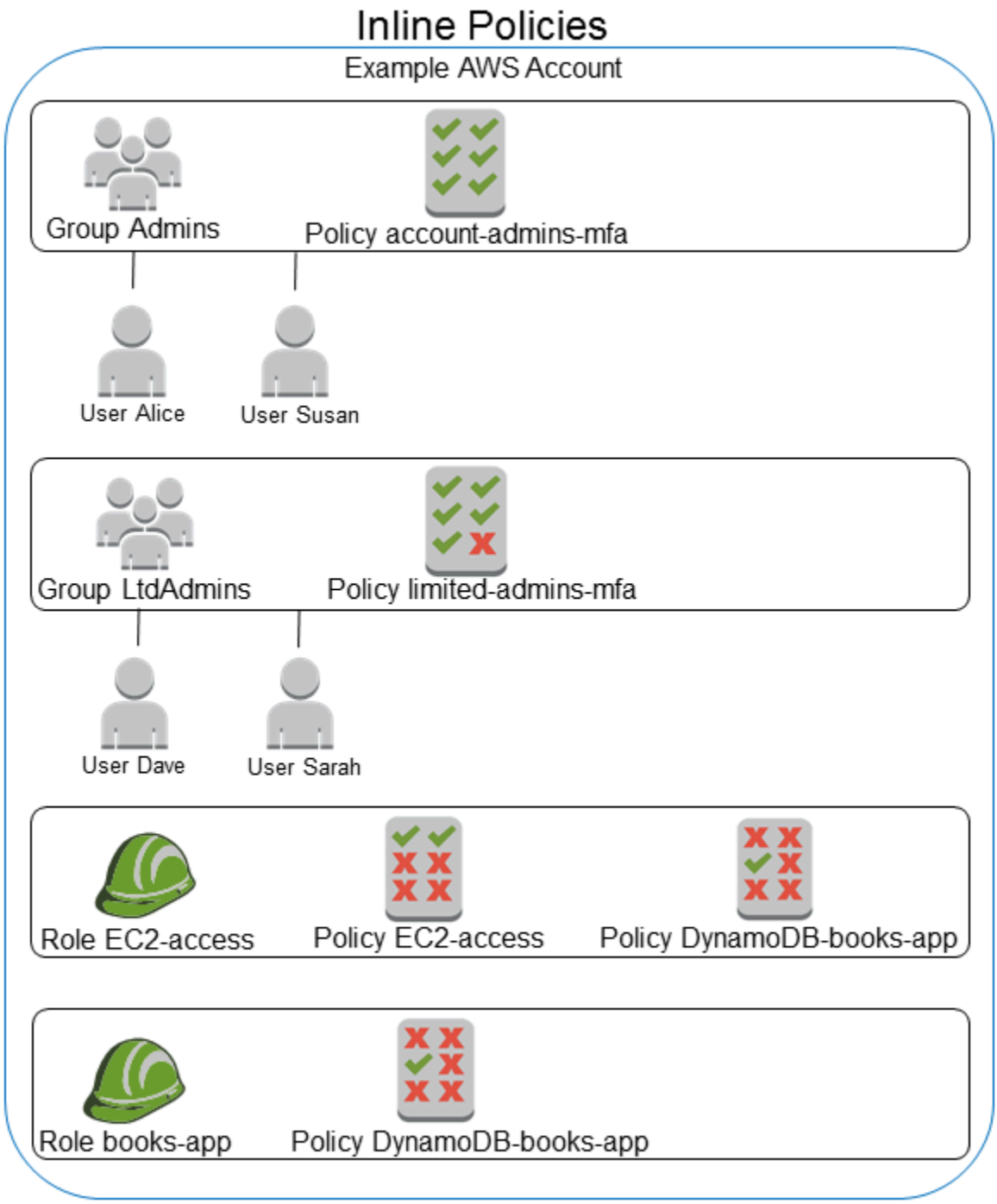
Untuk informasi selengkapnya, silakan lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan



Kebijakan inline

Kebijakan inline adalah kebijakan yang dibuat untuk satu IAM identitas (pengguna, grup pengguna, atau peran). Kebijakan inline mempertahankan one-to-one hubungan yang ketat antara kebijakan dan identitas. Mereka dihapus ketika Anda menghapus identitas. Anda dapat membuat kebijakan dan menyematkannya dalam identitas, baik saat Anda membuat identitas atau nanti. Jika kebijakan dapat berlaku untuk lebih dari satu entitas, lebih baik menggunakan kebijakan terkelola.

Diagram berikut menggambarkan kebijakan inline. Setiap kebijakan merupakan bagian yang melekat pada pengguna, grup, atau peran. Perhatikan bahwa dua peran menyertakan kebijakan yang sama (kebijakan DynamoDB-Books-App), tetapi mereka tidak membagikan kebijakan tunggal. Setiap peran memiliki salinan kebijakan sendiri.



Pilih antara kebijakan terkelola dan kebijakan inline

Pertimbangkan kasus penggunaan Anda saat memutuskan antara kebijakan terkelola dan sebaris. Dalam sebagian besar kasus, kami menyarankan agar Anda menggunakan kebijakan terkelola daripada kebijakan inline.

> **Note**
>
> Anda dapat menggunakan kebijakan terkelola dan sebaris bersama-sama untuk menentukan izin umum dan unik untuk entitas utama.

Kebijakan terkelola menyediakan fitur berikut:

Dapat digunakan kembali

Satu kebijakan terkelola dapat dilampirkan pada beberapa entitas prinsipal (pengguna, grup, dan peran). Anda dapat membuat pustaka kebijakan yang menentukan izin berguna untuk Anda Akun AWS, lalu melampirkan kebijakan ini ke entitas utama sesuai kebutuhan.

Manajemen perubahan pusat

Ketika Anda mengubah kebijakan terkelola, perubahan diterapkan pada semua entitas prinsipal yang terkait dengan kebijakan tersebut. Misalnya, jika ingin menambahkan izin untuk yang baru AWS API, Anda dapat memperbarui kebijakan yang dikelola pelanggan atau mengaitkan kebijakan AWS terkelola untuk menambahkan izin. Jika Anda menggunakan kebijakan AWS terkelola, AWS perbarui kebijakan tersebut. Ketika kebijakan terkelola diperbarui, perubahan diterapkan ke semua entitas utama yang dilampirkan oleh kebijakan terkelola. Sebaliknya, untuk mengubah kebijakan inline, Anda harus mengedit secara individual setiap identitas yang berisi kebijakan inline. Misalnya, jika grup dan peran keduanya berisi kebijakan inline yang sama, Anda harus mengedit kedua entitas utama secara individual untuk mengubah kebijakan tersebut.

Versioning dan peluncuran kembali

Ketika Anda mengubah kebijakan yang dikelola pelanggan, perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola. IAMmenyimpan hingga lima versi kebijakan yang dikelola pelanggan Anda. Anda dapat menggunakan versi kebijakan untuk mengembalikan kebijakan ke versi sebelumnya sesuai kebutuhan.

> **Note**
>
> Versi kebijakan berbeda dari elemen kebijakan `Version`. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Untuk mempelajari lebih lanjut tentang versi kebijakan, lihat the section called “Kebijakan IAM versioning”.

Untuk mempelajari lebih lanjut tentang elemen kebijakan `Version`, lihat IAMJSONelemen kebijakan: Version.

Menyerahkan manajemen perizinan

Anda dapat mengizinkan pengguna Akun AWS untuk melampirkan dan melepaskan kebijakan sambil mempertahankan kendali atas izin yang ditentukan dalam kebijakan tersebut. Untuk melakukan ini, tetapkan beberapa pengguna sebagai administrator lengkap—yaitu administrator yang dapat membuat, memperbarui, dan menghapus kebijakan. Kemudian Anda dapat menunjuk pengguna lain sebagai administrator terbatas. Administrator terbatas tersebut dapat melampirkan kebijakan ke entitas utama lainnya, tetapi hanya kebijakan yang Anda izinkan untuk dilampirkan.

Untuk informasi lebih lanjut tentang menyerahkan manajemen perizinan, lihat Mengendalikan akses ke kebijakan.

Batas karakter kebijakan yang lebih besar

Batas ukuran karakter maksimum untuk kebijakan terkelola lebih besar dari batas karakter untuk kebijakan sebaris grup. Jika Anda mencapai batas ukuran karakter kebijakan sebaris, Anda dapat membuat IAM grup lainnya dan melampirkan kebijakan terkelola ke grup.

Untuk informasi lebih lanjut tentang kuota dan batas, lihatIAMdan AWS STS kuota.

Pembaruan otomatis untuk kebijakan AWS terkelola

AWS mempertahankan kebijakan AWS terkelola dan memperbaruinya bila perlu, misalnya, untuk menambahkan izin untuk AWS layanan baru, tanpa Anda harus membuat perubahan. Pembaruan secara otomatis diterapkan ke entitas utama tempat Anda melampirkan kebijakan AWS terkelola.

Memulai kebijakan terkelola

Sebaiknya gunakan kebijakan yang memberikan hak istimewa paling sedikit, atau hanya memberikan izin yang diperlukan untuk melakukan tugas. Cara paling aman untuk memberikan hak istimewa paling sedikit adalah dengan menulis kebijakan yang dikelola pelanggan hanya dengan izin yang diperlukan oleh tim Anda. Anda harus membuat proses untuk memungkinkan tim Anda meminta lebih banyak izin bila diperlukan. Butuh waktu dan keahlian untuk membuat kebijakan terkelola IAM pelanggan yang hanya memberi tim Anda izin yang mereka butuhkan.

Untuk mulai menambahkan izin ke IAM identitas Anda (pengguna, grup pengguna, dan peran), Anda dapat menggunakannya. AWS kebijakan terkelola AWS kebijakan terkelola tidak memberikan

izin hak istimewa paling sedikit. Anda harus mempertimbangkan risiko keamanan memberikan izin kepada kepala sekolah Anda lebih banyak daripada yang mereka butuhkan untuk melakukan pekerjaan mereka.

Anda dapat melampirkan kebijakan AWS terkelola, termasuk fungsi pekerjaan, ke IAM identitas apa pun. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Untuk beralih ke izin hak istimewa terkecil, Anda dapat menjalankan AWS Identity and Access Management Access Analyzer untuk memantau prinsipal dengan kebijakan terkelola. AWS Setelah mengetahui izin yang mereka gunakan, Anda dapat menulis atau membuat kebijakan yang dikelola pelanggan hanya dengan izin yang diperlukan untuk tim Anda. Ini kurang aman, tetapi memberikan lebih banyak fleksibilitas saat Anda mempelajari bagaimana tim Anda menggunakan AWS. Untuk informasi selengkapnya, lihat IAMPembuatan kebijakan Access Analyzer.

AWS kebijakan terkelola dirancang untuk memberikan izin untuk banyak kasus penggunaan umum. Untuk informasi selengkapnya tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

Untuk daftar kebijakan AWS terkelola, lihat Panduan Referensi Kebijakan AWS Terkelola.

Menggunakan kebijakan inline

Kebijakan inline berguna jika Anda ingin mempertahankan one-to-one hubungan yang ketat antara kebijakan dan identitas yang diterapkan. Misalnya, jika Anda ingin memastikan bahwa izin dalam kebijakan tidak secara tidak sengaja ditetapkan ke identitas selain identitas yang dimaksudkan. Ketika Anda menggunakan kebijakan inline, izin dalam kebijakan tersebut tidak dapat secara tidak sengaja dilampirkan pada identitas yang salah. Selain itu, ketika Anda menggunakan AWS Management Console untuk menghapus identitas itu, kebijakan yang disematkan dalam identitas juga dihapus karena mereka adalah bagian dari entitas utama.

Mengonversi kebijakan inline menjadi kebijakan terkelola

Jika Anda memiliki kebijakan inline di akun Anda, Anda dapat mengonversinya ke kebijakan terkelola. Untuk melakukannya, salin kebijakan tersebut ke kebijakan baru yang dikelola. Selanjutnya, lampirkan kebijakan baru ke identitas yang memiliki kebijakan inline. Kemudian hapus kebijakan inline.

Mengonversi kebijakan inline menjadi kebijakan terkelola

Untuk mengonversi kebijakan inline ke kebijakan terkelola

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, atauPeran.
3. Dari daftar, pilih nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
4. Pilih tab Izin.
5. Untuk IAM grup, pilih nama kebijakan sebaris yang ingin Anda hapus. Untuk pengguna dan peran, pilih Tampilkan ***n*** lebih, jika perlu, dan kemudian memperluas kebijakan inline yang ingin Anda hapus.
6. Pilih Salin untuk menyalin dokumen JSON kebijakan untuk kebijakan tersebut.
7. Di panel navigasi, pilih Kebijakan.
8. Pilih Buat kebijakan dan kemudian pilih JSONopsi.
9. Ganti teks yang ada dengan teks JSON kebijakan Anda, lalu pilih Berikutnya.
10. Masukkan nama dan deskripsi opsional untuk kebijakan Anda dan pilih Buat kebijakan.
11. Di panel navigasi, pilih Grup pengguna, Pengguna, atau Peran, lalu pilih kembali nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
12. Pilih tab Izin dan kemudian pilih Tambahkan izin.
13. Untuk IAM grup, pilih kotak centang di samping nama kebijakan baru, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan. Untuk pengguna atau peran, pilih Tambahkan izin. Di halaman berikutnya, pilih Lampirkan kebijakan yang ada secara langsung, pilih kotak centang di samping nama kebijakan baru Anda, pilih Berikutnya, lalu pilih Tambahkan izin.

    Anda kembali ke halaman Ringkasan untuk grup pengguna, pengguna, atau peran Anda.
14. Pilih kotak centang di samping kebijakan sebaris yang ingin Anda hapus dan pilih Hapus.

Kebijakan terkelola yang tidak digunakan lagi AWS

Untuk menyederhanakan penetapan izin, AWS berikan kebijakan terkelola —kebijakan yang telah ditentukan sebelumnya yang siap dilampirkan ke pengguna, grup, dan peran IAM Anda.

Terkadang AWS perlu menambahkan izin baru ke kebijakan yang ada, seperti ketika layanan baru diperkenalkan. Menambahkan izin baru ke kebijakan yang sudah ada tidak akan mengganggu atau menghilangkan fitur atau kemampuan.

Namun, AWS mungkin memilih untuk membuat kebijakan baru ketika perubahan yang diperlukan dapat berdampak pada pelanggan jika diterapkan pada kebijakan yang ada. Misalnya, menghapus izin dari kebijakan yang sudah ada dapat merusak izin dari suatu entitas atau aplikasi IAM yang bergantung di atasnya, yang berpotensi mengganggu operasi yang bersifat kritis.

Oleh karena itu, ketika perubahan seperti itu diperlukan, AWS buat kebijakan yang sama sekali baru dengan perubahan yang diperlukan dan membuatnya tersedia untuk pelanggan. Kebijakan lama selanjutnya ditandai usang. Kebijakan terkelola usang muncul dengan ikon peringatan di sebelahnya dalam daftar Kebijakan di konsol IAM.

Kebijakan usang memiliki karakteristik sebagai berikut:

- Itu terus berfungsi untuk semua pengguna, grup, dan peran yang dilampirkan saat ini. Tidak ada yang rusak.
- Itu tidak dapat dilampirkan pada pengguna, grup, atau peran baru. Jika Anda melepasnya dari entri yang ada, Anda tidak dapat melampirkannya kembali.
- Setelah Anda melepasnya dari semua entri saat ini, itu tidak lagi terlihat dan tidak lagi dapat digunakan dengan cara apa pun.

Jika pengguna, grup, atau peran membutuhkan kebijakan tersebut, Anda harus melampirkan kebijakan baru. Saat Anda menerima pemberitahuan bahwa kebijakan telah usang, kami menyarankan agar Anda segera merencanakan untuk melampirkan semua pengguna, grup, dan peran ke kebijakan pengganti dan memisahkannya dari kebijakan yang telah usang tersebut. Terus menggunakan kebijakan yang sudah usang dapat membawa risiko yang hanya dapat dikurangi dengan melakukan pergantian ke kebijakan pengganti.

Tetapkan pagar pembatas izin menggunakan perimeter data

Pagar perimeter data dimaksudkan untuk berfungsi sebagai batas yang selalu aktif untuk membantu melindungi data Anda di seluruh rangkaian akun dan sumber daya yang luas. AWS Perimeter data mengikuti praktik terbaik keamanan IAM untuk membuat pagar pembatas izin di beberapa akun. Pagar pembatas izin di seluruh organisasi ini tidak menggantikan kontrol akses berbutir halus yang ada. Sebaliknya, mereka bekerja sebagai kontrol akses kasar yang membantu meningkatkan strategi

keamanan Anda dengan memastikan pengguna, peran, dan sumber daya mematuhi serangkaian standar keamanan yang ditetapkan.

Perimeter data adalah set pagar pembatas izin di AWS lingkungan Anda yang membantu memastikan bahwa hanya identitas tepercaya Anda yang mengakses sumber daya tepercaya dari jaringan yang diharapkan.

- Identitas tepercaya: Prinsipal (peran atau pengguna IAM) di AWS akun dan AWS layanan Anda yang bertindak atas nama Anda.
- Sumber daya tepercaya: Sumber daya yang dimiliki oleh AWS akun Anda atau oleh AWS layanan yang bertindak atas nama Anda.
- Jaringan yang diharapkan: Pusat data lokal Anda dan awan pribadi virtual (VPC), atau jaringan AWS layanan yang bertindak atas nama Anda.

> **Note**
>
> Dalam beberapa kasus, Anda mungkin perlu memperluas perimeter data Anda untuk juga menyertakan akses oleh mitra bisnis tepercaya Anda. Anda harus mempertimbangkan semua pola akses data yang dimaksudkan ketika Anda membuat definisi identitas tepercaya, sumber daya tepercaya, dan jaringan yang diharapkan khusus untuk perusahaan Anda dan penggunaan Layanan AWS Anda.

Kontrol perimeter data harus diperlakukan sebagai kontrol keamanan lainnya dalam program keamanan informasi dan manajemen risiko. Ini berarti bahwa Anda harus melakukan analisis ancaman untuk mengidentifikasi potensi risiko dalam lingkungan cloud Anda, dan kemudian, berdasarkan kriteria penerimaan risiko Anda sendiri, pilih dan terapkan kontrol perimeter data yang sesuai. Untuk lebih menginformasikan pendekatan berbasis risiko berulang untuk implementasi perimeter data, Anda perlu memahami risiko keamanan dan vektor ancaman apa yang ditangani oleh kontrol perimeter data serta prioritas keamanan Anda.

Kontrol perimeter data

Kontrol berbutir kasar perimeter data membantu Anda mencapai enam tujuan keamanan yang berbeda di tiga perimeter data melalui penerapan kombinasi yang berbeda dari dan kunci kondisi. Jenis kebijakan

Perimeter	Tujuan kontrol	Penggunaan	Diterapkan pada	Kunci konteks kondisi global
Identitas	Hanya identitas tepercaya yang dapat mengakses sumber daya saya	Kebijakan berbasis sumber daya	Sumber daya	aws: PrincipalOrg ID aws: PrincipalOrgPaths aws: PrincipalAccount aws: PrincipalIsAwsService
	Hanya identitas tepercaya yang diizinkan dari jaringan saya	Kebijakan titik akhir VPC	Jaringan	
Sumber daya	Identitas Anda hanya dapat mengakses sumber daya tepercaya	SCP	Identitas	aws: ResourceOrg ID aws: ResourceOrgPaths aws: ResourceAccount
	Hanya sumber daya tepercaya yang dapat diakses dari jaringan Anda	Kebijakan titik akhir VPC	Jaringan	
Jaringan	Identitas Anda hanya dapat mengakses sumber daya dari jaringan yang diharapkan	SCP	Identitas	aws: SourceIp aws: SourceVpc aws: SourceVpce AWS: melalui AWSService aws: PrincipalIsAwsService
	Sumber daya Anda hanya dapat diakses	Kebijakan berbasis sumber daya	Sumber daya	

Perimeter	Tujuan kontrol	Penggunaan	Diterapkan pada	Kunci konteks kondisi global
	dari jaringan yang diharapkan			

Anda dapat menganggap perimeter data sebagai membuat batas yang kuat di sekitar data Anda untuk mencegah pola akses yang tidak diinginkan. Meskipun perimeter data dapat mencegah akses luas yang tidak diinginkan, Anda masih perlu membuat keputusan kontrol akses yang halus. Membuat perimeter data tidak mengurangi kebutuhan untuk terus menyempurnakan izin dengan menggunakan alat seperti IAM Access Analyzer sebagai bagian dari perjalanan Anda menuju hak istimewa yang paling sedikit.

Perimeter identitas

Perimeter identitas adalah serangkaian kontrol akses pencegahan kasar yang membantu memastikan hanya identitas tepercaya yang dapat mengakses sumber daya Anda dan hanya identitas tepercaya yang diizinkan dari jaringan Anda. Identitas tepercaya mencakup kepala sekolah (peran atau pengguna) di AWS akun dan AWS layanan Anda yang bertindak atas nama Anda. Semua identitas lainnya dianggap tidak dipercaya dan dicegah oleh perimeter identitas kecuali pengecualian eksplisit diberikan.

Kunci kondisi global berikut membantu menegakkan kontrol perimeter identitas. Gunakan kunci ini dalam kebijakan berbasis sumber daya untuk membatasi akses ke sumber daya, atau dalam kebijakan titik akhir VPC untuk membatasi akses ke jaringan Anda.

- aws: PrincipalOrg ID— Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa kepala sekolah IAM yang membuat permintaan milik organisasi yang ditentukan di. AWS Organizations
- aws: PrincipalOrgPaths— Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa pengguna IAM, peran IAM, pengguna federasi, atau A yang Pengguna root akun AWS membuat permintaan milik unit organisasi tertentu (OU) di. AWS Organizations
- aws: PrincipalAccount— Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya hanya dapat diakses oleh akun utama yang Anda tentukan dalam kebijakan.
- aws: PrincipalIs AWSServicedan aws: SourceOrg ID (secara bergantian aws: SourceOrgPaths danaws: SourceAccount) — Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa

ketika Layanan AWS kepala sekolah mengakses sumber daya Anda, mereka melakukannya hanya atas nama sumber daya di organisasi tertentu, unit organisasi, atau akun di. AWS Organizations

Untuk informasi selengkapnya, lihat Membuat perimeter data di AWS: Izinkan hanya identitas tepercaya untuk mengakses data perusahaan.

Perimeter sumber daya

Perimeter sumber daya adalah serangkaian kontrol akses pencegahan kasar yang membantu memastikan identitas Anda hanya dapat mengakses sumber daya tepercaya dan hanya sumber daya tepercaya yang dapat diakses dari jaringan Anda. Sumber daya tepercaya mencakup sumber daya yang dimiliki oleh AWS akun Anda atau oleh AWS layanan yang bertindak atas nama Anda.

Kunci kondisi global berikut membantu menegakkan kontrol perimeter sumber daya. Gunakan kunci ini dalam Kebijakan Kontrol Layanan (SCP) untuk membatasi sumber daya mana yang dapat diakses oleh identitas Anda, atau dalam kebijakan titik akhir VPC untuk membatasi sumber daya mana yang dapat diakses dari jaringan Anda.

- aws: ResourceOrg ID— Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya yang sedang diakses milik organisasi tertentu di AWS Organizations.
- aws: ResourceOrgPaths— Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya yang sedang diakses milik unit organisasi tertentu (OU) di AWS Organizations.
- aws: ResourceAccount— Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya yang sedang diakses milik akun yang ditentukan di AWS Organizations.

Dalam beberapa kasus, Anda mungkin perlu mengizinkan akses ke sumber daya yang AWS dimiliki, sumber daya yang bukan milik organisasi Anda dan yang diakses oleh kepala sekolah Anda atau oleh AWS layanan yang bertindak atas nama Anda. Untuk informasi selengkapnya tentang skenario ini, lihat Membuat perimeter data di AWS: Izinkan hanya sumber daya tepercaya dari organisasi saya.

Perimeter jaringan

Perimeter jaringan adalah seperangkat kontrol akses pencegahan kasar yang membantu memastikan identitas Anda dapat mengakses sumber daya hanya dari jaringan yang diharapkan dan sumber daya Anda hanya dapat diakses dari jaringan yang diharapkan. Jaringan yang diharapkan mencakup pusat data lokal dan virtual private cloud (VPC) dan jaringan AWS layanan yang bertindak atas nama Anda.

Kunci kondisi global berikut membantu menegakkan kontrol perimeter jaringan. Gunakan kunci ini dalam Kebijakan Kontrol Layanan (SCP) untuk membatasi jaringan yang dapat dikomunikasikan oleh identitas Anda, atau dalam kebijakan berbasis sumber daya untuk membatasi akses sumber daya ke jaringan yang diharapkan.

- aws: SourceIp— Anda dapat menggunakan kunci kondisi ini untuk memastikan alamat IP pemohon berada dalam rentang IP tertentu.
- aws: SourceVpc— Anda dapat menggunakan kunci kondisi ini untuk memastikan titik akhir VPC yang dilalui permintaan milik VPC yang ditentukan.
- aws: SourceVpce— Anda dapat menggunakan kunci kondisi ini untuk memastikan permintaan berjalan melalui titik akhir VPC yang ditentukan.
- AWS: V iaAWSService— Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa Layanan AWS dapat membuat permintaan atas nama kepala sekolah Anda menggunakan Teruskan sesi akses (FAS).
- aws: PrincipalIs AWSService— Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa Layanan AWS dapat mengakses sumber daya Anda menggunakanAWS prinsipal layanan.

Ada skenario tambahan di mana Anda perlu mengizinkan akses ke Layanan AWS sumber daya Anda dari luar jaringan Anda. Untuk informasi selengkapnya, lihat Membuat perimeter data pada AWS: Izinkan akses ke data perusahaan hanya dari jaringan yang diharapkan.

Sumber daya untuk mempelajari lebih lanjut tentang perimeter data

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang perimeter data. AWS

- Perimeter data aktif AWS— Pelajari tentang perimeter data dan manfaat serta kasus penggunaannya.
- Whitepaper: Membangun Perimeter Data pada AWS — Paper ini menguraikan praktik terbaik dan layanan yang tersedia untuk membuat perimeter di sekitar identitas, sumber daya, dan jaringan Anda. AWS
- Webinar: Membangun perimeter data di AWS - Pelajari di mana dan bagaimana menerapkan kontrol perimeter data berdasarkan skenario risiko yang berbeda.
- Seri Posting Blog: Membangun Perimeter Data pada AWS — Posting blog ini mencakup panduan preskriptif tentang menetapkan perimeter data Anda dalam skala besar, termasuk pertimbangan keamanan dan implementasi utama.

- Contoh kebijakan perimeter data — GitHub Repositori ini berisi contoh kebijakan yang mencakup beberapa pola umum untuk membantu Anda menerapkan perimeter data. AWS
- Pembantu perimeter data — Alat ini membantu Anda merancang dan mengantisipasi dampak kontrol perimeter data Anda dengan menganalisis aktivitas akses di log Anda AWS CloudTrail.

Batas izin untuk entitas IAM

AWS mendukung batas izin untuk IAM entitas (pengguna atau peran). Batas izin adalah fitur lanjutan untuk menggunakan kebijakan terkelola guna menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas. IAM Batas izin entitas mengizinkannya untuk melakukan hanya tindakan yang diizinkan oleh kebijakan berbasis identitas dan batas izinnya.

Untuk informasi lebih lanjut tentang jenis kebijakan, lihat Jenis kebijakan.

> **Important**
>
> Jangan gunakan pernyataan kebijakan berbasis sumber daya yang menyertakan elemen `NotPrincipal` kebijakan dengan `Deny` efek bagi IAM pengguna atau peran yang memiliki kebijakan batas izin yang dilampirkan. `NotPrincipal`Elemen dengan `Deny` efek akan selalu menolak IAM prinsip apa pun yang memiliki kebijakan batas izin yang dilampirkan, terlepas dari nilai yang ditentukan dalam elemen. `NotPrincipal` Hal ini menyebabkan beberapa IAM pengguna atau peran yang seharusnya memiliki akses ke sumber daya kehilangan akses. Sebaiknya ubah pernyataan kebijakan berbasis sumber daya Anda untuk menggunakan operator kondisi `ArnNotEquals`dengan kunci `aws:PrincipalArn`konteks untuk membatasi akses, bukan elemen. `NotPrincipal` Untuk informasi tentang `NotPrincipal` elemen, lihatAWS JSONelemen kebijakan: NotPrincipal.

Anda dapat menggunakan kebijakan AWS terkelola atau kebijakan yang dikelola pelanggan untuk menetapkan batas IAM entitas (pengguna atau peran). Kebijakan tersebut membatasi izin maksimum bagi pengguna atau peran.

Misalnya, asumsikan bahwa nama IAM pengguna `ShirleyRodriguez` harus diizinkan untuk mengelola hanya Amazon S3, Amazon CloudWatch, dan Amazon. EC2 Untuk menegakkan aturan ini, Anda dapat menggunakan kebijakan berikut untuk mengatur batas izin untuk pengguna `ShirleyRodriguez`:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Saat Anda menggunakan kebijakan untuk mengatur batas izin bagi pengguna, itu membatasi izin pengguna tetapi tidak memberikan izin sendiri. Dalam contoh ini, kebijakan menetapkan izin maksimum `ShirleyRodriguez` sebagai semua operasi di Amazon S3 CloudWatch, dan Amazon. EC2 Shirley tidak pernah dapat melakukan operasi di layanan lain, termasukIAM, bahkan jika dia memiliki kebijakan izin yang mengizinkannya. Misalnya, Anda dapat menambahkan kebijakan berikut ke pengguna `ShirleyRodriguez`:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Kebijakan ini memungkinkan pembuatan pengguna diIAM. Jika Anda melampirkan kebijakan izin ini ke pengguna `ShirleyRodriguez` dan Shirley mencoba membuat sebuah pengguna, operasi gagal. Itu gagal karena batas izin tidak mengizinkan operasi `iam:CreateUser`. Mengingat kedua kebijakan ini, Shirley tidak memiliki izin untuk melakukan operasi apa pun di AWS. Anda harus menambahkan kebijakan izin yang berbeda untuk mengizinkan tindakan di layanan lain, seperti Amazon S3. Atau, Anda dapat memperbarui batas izin untuk memungkinkannya membuat pengguna masuk. IAM

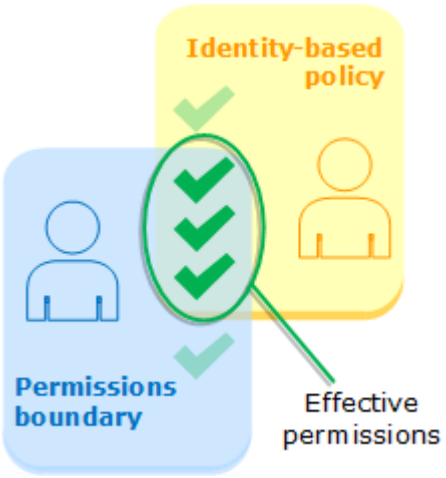
Mengevaluasi izin efektif dengan batasan

Batas izin untuk IAM entitas (pengguna atau peran) menetapkan izin maksimum yang dapat dimiliki entitas. Ini dapat mengubah izin efektif bagi pengguna atau peran tersebut. Izin efektif untuk

sebuah entitas adalah izin yang diberikan oleh semua kebijakan yang memengaruhi pengguna atau peran. Dalam akun, izin untuk entitas dapat dipengaruhi oleh kebijakan berbasis identitas, kebijakan berbasis sumber daya, batas izin, Organisasi, atau kebijakan sesi. SCPs Untuk informasi selengkapnya tentang berbagai jenis kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

Jika salah satu dari jenis kebijakan ini secara jelas menolak akses untuk suatu operasi, maka permintaan tersebut ditolak. Izin yang diberikan ke suatu entitas oleh beberapa jenis izin jauh lebih rumit. Untuk detail selengkapnya tentang cara AWS mengevaluasi kebijakan, lihatLogika evaluasi kebijakan.

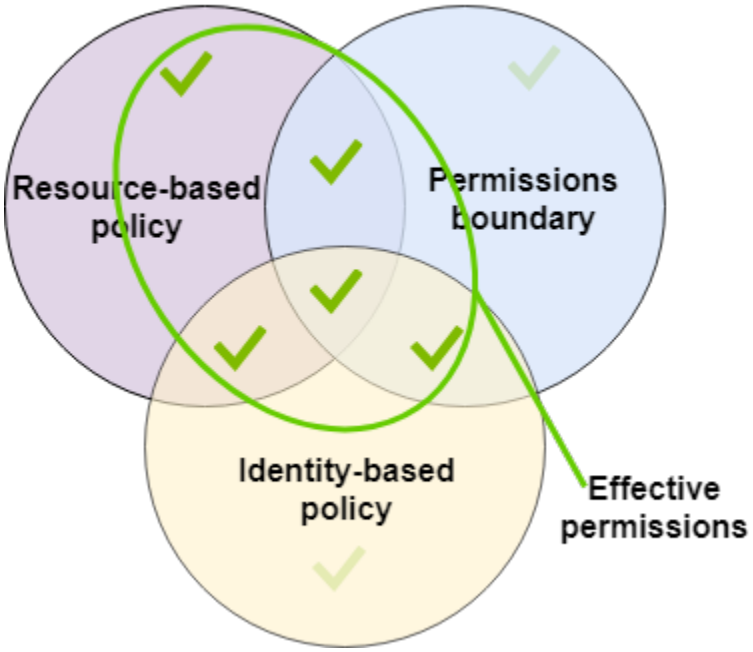
Kebijakan berbasis identitas dengan batasan – Kebijakan berbasis identitas bersifat inline atau kebijakan terkelola yang dilampirkan pada pengguna, grup pengguna, atau peran. Kebijakan berbasis identitas memberikan izin kepada sebuah entitas, dan batasan izin membatasi izin tersebut. Izin yang efektif adalah titik temu dari kedua jenis kebijakan. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.



Kebijakan berbasis sumber daya – Kebijakan berbasis sumber daya mengontrol bagaimana prinsipal tertentu dapat mengakses sumber daya tempat kebijakan tersebut dilampirkan.

Kebijakan berbasis sumber daya untuk pengguna IAM

> Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada IAM pengguna ARN (yang bukan sesi pengguna gabungan) tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas atau batas izin.



Kebijakan berbasis sumber daya untuk peran IAM

IAMperan — Kebijakan berbasis sumber daya yang memberikan izin untuk IAM peran ARN dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

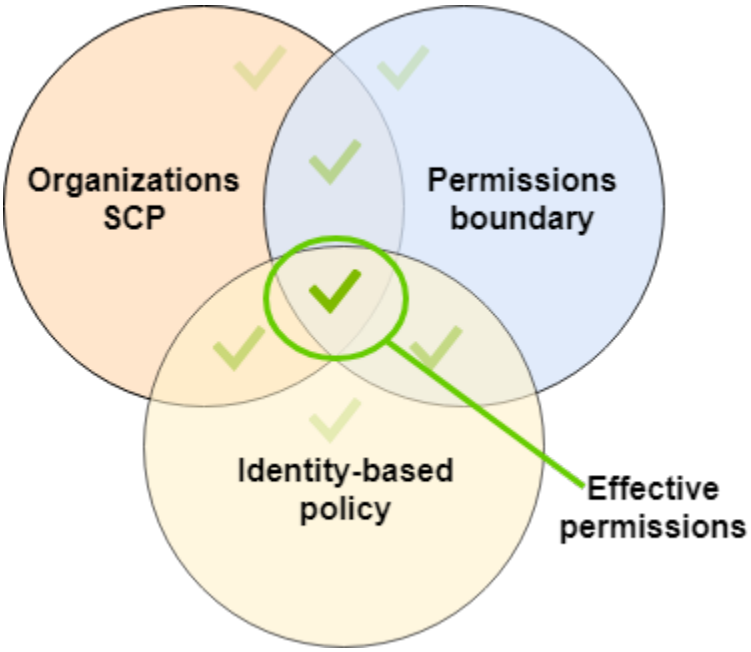
IAMsesi peran — Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin untuk sesi peran memberikan ARN izin langsung ke sesi IAM peran yang diasumsikan. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi. Ketika Anda mengambil peran dan membuat permintaan, kepala sekolah yang membuat permintaan adalah sesi IAM peran ARN dan bukan peran itu sendiri. ARN

Kebijakan berbasis sumber daya untuk sesi pengguna federasi IAM

IAMsesi pengguna federasi — Sesi pengguna IAM federasi adalah sesi yang dibuat dengan menelepon. `GetFederationToken` Ketika pengguna federasi membuat permintaan, prinsipal yang membuat permintaan adalah pengguna federasi ARN dan bukan IAM pengguna yang ARN berfederasi. Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada pengguna ARN federasi memberikan izin langsung ke sesi. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi.

Namun, jika kebijakan berbasis sumber daya memberikan izin kepada pengguna yang melakukan ARN federasi, maka permintaan yang dibuat oleh IAM pengguna federasi selama sesi dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

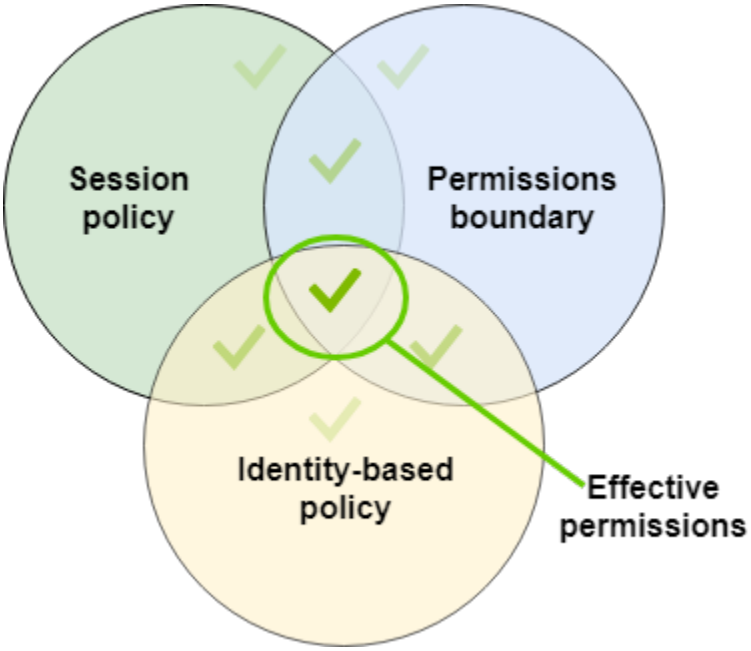
Organizations SCPs — SCPs diterapkan untuk keseluruhan Akun AWS. Mereka membatasi izin untuk setiap permintaan yang dibuat oleh prinsipal di dalam akun. IAMEntitas (pengguna atau peran) dapat membuat permintaan yang dipengaruhi olehSCP, batas izin, dan kebijakan berbasis identitas. Dalam hal ini, permintaan hanya diizinkan jika ketiga jenis kebijakan mengizinkannya. Izin yang efektif adalah titik temu dari ketiga jenis kebijakan. Penolakan secara eksplisit dalam salah satu kebijakan ini membatalkan izin.



Anda dapat mempelajari apakah akun Anda adalah anggota organisasi di AWS Organizations. Anggota organisasi mungkin terpengaruh oleh suatuSCP. Untuk melihat data ini menggunakan AWS CLI perintah atau AWS API operasi, Anda harus memiliki izin untuk `organizations:DescribeOrganization` tindakan untuk entitas Organizations Anda. Anda harus memiliki izin tambahan untuk melakukan operasi di konsol Organisasi. Untuk mengetahui SCP apakah menolak akses ke permintaan tertentu, atau untuk mengubah izin efektif Anda, hubungi administrator Anda AWS Organizations .

Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin untuk sesi berasal dari IAM entitas (pengguna atau peran) yang digunakan untuk membuat sesi dan dari kebijakan sesi. Izin kebijakan berbasis identitas milik entitas akan dibatasi oleh kebijakan

sesi dan batasan izin. Izin yang efektif untuk set jenis kebijakan ini adalah titik temu dari ketiga jenis kebijakan. Penolakan secara eksplisit dalam salah satu kebijakan ini membatalkan izin. Untuk informasi selengkapnya tentang kebijakan sesi, lihat Kebijakan Sesi



Mendelegasikan tanggung jawab kepada orang lain menggunakan batas izin

Anda dapat menggunakan batas izin untuk mendelegasikan tugas pengelolaan izin, seperti pembuatan pengguna, kepada IAM pengguna di akun Anda. Hal ini mengizinkan orang lain melakukan tugas atas nama Anda dalam batas izin tertentu.

Misalnya, asumsikan bahwa María adalah administrator Akun AWS X-Company. Dia ingin memberikanan tugas pembuatan pengguna ke Zhang. Namun, ia harus memastikan bahwa Zhang membuat pengguna yang mematuhi aturan perusahaan berikut:

- Pengguna tidak dapat menggunakan IAM untuk membuat atau mengelola pengguna, grup, peran, atau kebijakan.
- Pengguna ditolak akses ke `logs` bucket Amazon S3 dan tidak dapat mengakses instans `i-1234567890abcdef0` AmazonEC2.
- Pengguna tidak dapat menghapus kebijakan batas milik mereka.

Untuk menegakkan aturan ini, María menyelesaikan tugas berikut, yang perinciannya tercantum di bawah ini:

1. María membuat kebijakan terkelola `XCompanyBoundaries` untuk digunakan sebagai batas izin bagi semua pengguna baru dalam akun.
2. María membuat kebijakan terkelola `DelegatedUserBoundary` dan menetapkannya sebagai batas izin untuk Zhang. Maria membuat catatan tentang pengguna adminnya ARN dan menggunakannya dalam kebijakan untuk mencegah Zhang mengaksesnya.
3. María membuat kebijakan terkelola `DelegatedUserPermissions` dan melampirkannya sebagai kebijakan izin untuk Zhang.
4. María memberi tahu Zhang tentang tanggung jawab dan batasan barunya.

Tugas 1: María harus terlebih dahulu membuat kebijakan terkelola untuk menentukan batas bagi pengguna baru. María akan mengizinkan Zhang untuk memberikan kebijakan izin yang dibutuhkan pengguna, tetapi dia ingin pengguna tersebut dibatasi. Untuk melakukan ini, ia membuat kebijakan terkelola pelanggan berikut ini dengan nama `XCompanyBoundaries`. Kebijakan ini melakukan hal-hal berikut:

- Mengizinkan pengguna akses penuh ke beberapa layanan
- Memungkinkan akses pengelolaan mandiri terbatas di IAM konsol. Ini berarti mereka dapat mengubah kata sandi mereka setelah masuk ke konsol. Mereka tidak dapat mengatur kata sandi awal mereka. Untuk mengijinkan ini, tambahkan tindakan `"*LoginProfile"` ke pernyataan `AllowManageOwnPasswordAndAccessKeys`.
- Menolak akses pengguna ke bucket log Amazon S3 atau instans Amazon `i-1234567890abcdef0` EC2

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
            ],
            "Resource": "*"
        },
```

```
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

Setiap pernyataan memiliki tujuan yang berbeda:

1. `ServiceBoundaries`Pernyataan kebijakan ini memungkinkan akses penuh ke AWS layanan yang ditentukan. Artinya jika tindakan pengguna baru dalam layanan ini hanya dibatasi oleh kebijakan izin yang terlampir pada pengguna.
2. `AllowIAMConsoleForCredentials`Pernyataan ini memungkinkan akses untuk mencantumkan semua IAM pengguna. Akses ini diperlukan untuk menavigasi halaman Pengguna di AWS Management Console. Ini juga mengijinkan untuk melihat persyaratan kata sandi untuk akun, yang diperlukan saat mengubah kata sandi Anda sendiri.
3. `AllowManageOwnPasswordAndAccessKeys`Pernyataan ini memungkinkan pengguna untuk mengelola hanya kata sandi konsol mereka sendiri dan kunci akses terprogram. Ini penting jika Zhang atau administrator lain memberikan pengguna baru kebijakan izin dengan akses penuh. IAM Dalam kasus ini, pengguna tersebut kemudian dapat mengubah izin miliknya sendiri atau milik pengguna lain. Pernyataan ini mencegah hal tersebut terjadi.
4. Pernyataan `DenyS3Logs` secara jelas menolak akses ke bucket `logs`.
5. Pernyataan `DenyEC2Production` secara jelas menolak akses ke instans `i-1234567890abcdef0`.

Tugas 2: María ingin mengizinkan Zhang untuk membuat semua pengguna Perusahaan X, tetapi hanya dengan batas izin `XCompanyBoundaries`. Dia menciptakan kebijakan terkelola pelanggan berikut bernama `DelegatedUserBoundary`. Kebijakan ini menentukan izin maksimum yang dapat dimiliki Zhang.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
            "Condition": {
               "StringEquals": {
```

```
                "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
              "iam:GetGroup",
              "iam:GetLoginProfile",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam:GetRolePolicy",
              "iam:GetUser",
              "iam:GetUserPolicy",
              "iam:ListAccessKeys",
              "iam:ListAttachedRolePolicies",
              "iam:ListAttachedUserPolicies",
              "iam:ListEntitiesForPolicy",
              "iam:ListGroups",
              "iam:ListGroupsForUser",
              "iam:ListMFADevices",
              "iam:ListPolicies",
              "iam:ListPolicyVersions",
              "iam:ListRolePolicies",
              "iam:ListSSHPublicKeys",
              "iam:ListServiceSpecificCredentials",
              "iam:ListSigningCertificates",
              "iam:ListUserPolicies",
              "iam:ListUsers",
              "iam:SetDefaultPolicyVersion",
              "iam:SimulateCustomPolicy",
              "iam:SimulatePrincipalPolicy",
```

```
                "iam:UpdateGroup",
                "iam:UpdateLoginProfile",
                "iam:UpdateUser"
            ],
            "NotResource": "arn:aws:iam::123456789012:user/Maria"
        },
        {
            "Sid": "NoBoundaryPolicyEdit",
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:DeletePolicy",
                "iam:DeletePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
                "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
            ]
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

Setiap pernyataan memiliki tujuan yang berbeda:

1. `CreateOrChangeOnlyWithBoundary`Pernyataan tersebut memungkinkan Zhang untuk membuat IAM pengguna tetapi hanya jika dia menggunakan `XCompanyBoundaries` kebijakan untuk menetapkan batas izin. Pernyataan ini juga mengizinkannya untuk menetapkan batas izin bagi pengguna yang ada tetapi hanya menggunakan kebijakan yang sama. Terakhir, pernyataan ini memungkinkan Zhang untuk mengelola kebijakan izin bagi pengguna dengan set batas izin ini.
2. Pernyataan `CloudWatchAndOtherIAMTasks` mengizinkan Zhang untuk menyelesaikan tugas pengguna, grup, dan kebijakan lainnya. Dia memiliki izin untuk mengatur ulang kata sandi dan membuat kunci akses untuk setiap IAM pengguna yang tidak tercantum dalam elemen `NotResource` kebijakan. Ini memungkinkannya untuk membantu pengguna dengan masalah masuk.

3. Pernyataan `NoBoundaryPolicyEdit` menolak akses Zhang untuk memperbarui kebijakan `XCompanyBoundaries`. Dia tidak diperbolehkan mengubah kebijakan apa pun yang digunakan untuk menetapkan batas izin bagi dirinya sendiri atau pengguna lain.
4. Pernyataan `NoBoundaryUserDelete` menolak akses Zhang untuk menghapus batas izin bagi dirinya sendiri atau pengguna lain.

Kemudian María menetapkan kebijakan `DelegatedUserBoundary` sebagai batas izin untuk pengguna Zhang.

Tugas 3: Karena batas izin membatasi izin maksimum, tetapi tidak memberikan akses sendiri, Maria harus membuat kebijakan izin untuk Zhang. Dia menciptakan kebijakan berikut bernama `DelegatedUserPermissions`. Kebijakan ini menetapkan operasi yang dapat dilakukan Zhang, dalam batas izin yang ditentukan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3BucketContents",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
```

```
    ]
}
```

Setiap pernyataan memiliki tujuan yang berbeda:

1. `IAM`Pernyataan kebijakan memungkinkan Zhang akses penuh keIAM. Namun, karena batas izinnya hanya memungkinkan beberapa IAM operasi, IAM izin efektifnya hanya dibatasi oleh batas izinnya.
2. `CloudWatchLimited`Pernyataan itu memungkinkan Zhang untuk melakukan lima tindakan. CloudWatch Batas izinnya memungkinkan semua tindakan masuk CloudWatch, sehingga CloudWatch izin efektifnya hanya dibatasi oleh kebijakan izinnya.
3. Pernyataan `S3BucketContents` mengizinkan Zhang membuat daftar Amazon S3 bucket `ZhangBucket`. Namun, batas izinnya tidak mengizinkan tindakan Amazon S3 apapun, sehingga dia tidak dapat melakukan operasi S3 apapun, terlepas dari kebijakan perizinannya.

> **Note**
>
> Kebijakan Zhang mengizinkannya untuk membuat pengguna yang kemudian dapat mengakses sumber daya Amazon S3 yang tidak dapat diaksesnya. Dengan menyerahkan tindakan administratif ini, Maria secara efektif memercayai Zhang dengan akses ke Amazon S3

Kemudian María menetapkan kebijakan `DelegatedUserPermissions` sebagai kebijakan izin untuk pengguna `Zhang`.

Tugas 4: Dia memberikan Zhang instruksi untuk membuat pengguna baru. Dia mengatakan padanya bahwa dia dapat membuat pengguna baru dengan izin apapun yang mereka butuhkan, tetapi ia harus menetapkan untuknya kebijakan `XCompanyBoundaries` sebagai batas izin.

Zhang menyelesaikan tugas berikut:

1. Zhang menciptakan pengguna dengan. AWS Management Console Dia mengetik nama pengguna `Nikhil` dan mengaktifkan akses konsol untuk pengguna. Dia membersihkan kotak centang di sebelah Memerlukan pengaturan ulang kata sandi, karena kebijakan di atas memungkinkan pengguna untuk mengubah kata sandi mereka hanya setelah mereka masuk ke konsol. IAM
2. Pada halaman Setel izin, Zhang memilih kebijakan ReadOnlyAccess izin IAMFullAccessdan AmazonS3 yang memungkinkan Nikhil melakukan pekerjaannya.

3. Zhang melewatkan bagian Atur batasan izin, melupakan instruksi María.
4. Zhang meninjau detail pengguna dan memilih Buat pengguna.

   Operasi gagal dan akses ditolak. Batas izin Zhang `DelegatedUserBoundary` mengharuskan setiap pengguna yang dia buat memiliki kebijakan `XCompanyBoundaries` yang digunakan sebagai batas izin.
5. Zhang kembali ke halaman sebelumnya. Di bagian Atur batasan izin, ia memilih kebijakan `XCompanyBoundaries`.
6. Zhang meninjau detail pengguna dan memilih Buat pengguna.

   Pengguna dibuat.

Ketika Nikhil masuk, ia memiliki akses ke IAM dan Amazon S3, kecuali operasi yang ditolak oleh batas izin. Misalnya, dia dapat mengubah kata sandinya sendiri IAM tetapi tidak dapat membuat pengguna lain atau mengedit kebijakannya. Nikhil memiliki akses hanya baca ke Amazon S3

Jika seseorang menambahkan kebijakan berbasis sumber daya ke bucket `logs` yang mengizinkan Nikhil memasukkan sebuah objek ke dalam bucket, dia tetap tidak dapat mengakses bucket. Alasannya adalah bahwa tindakan apa pun di bucket `logs` secara jelas ditolak oleh batas izinnya. Penolakan secara jelas dalam jenis kebijakan apa pun menyebabkan permintaan ditolak. Namun, jika kebijakan berbasis sumber daya yang terlampir pada rahasia Secrets Manager memungkinkan Nikhil untuk melakukan tindakan `secretsmanager:GetSecretValue`, lalu Nikhil dapat mengambil dan mendekripsi rahasia. Alasannya adalah bahwa operasi Secrets Manager tidak secara jelas ditolak oleh batas izinnya, dan penolakan implisit dalam batas izin tidak membatasi kebijakan berbasis sumber daya.

Kebijakan berbasis identitas dan kebijakan berbasis sumber daya

Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. Saat Anda membuat kebijakan izin untuk membatasi akses ke sumber daya, Anda dapat memilih kebijakan berbasis identitas atau kebijakan berbasis sumber daya.

Kebijakan berbasis identitas terlampir pada pengguna, grup, atau peran IAM. Kebijakan ini memungkinkan Anda menentukan apa yang dapat dilakukan oleh identitas (izinnya). Misalnya, Anda dapat melampirkan kebijakan ke pengguna IAM bernama John, yang menyatakan bahwa dia diizinkan untuk melakukan tindakan `RunInstances` Amazon EC2. Kebijakan selanjutnya dapat menyatakan bahwa John diizinkan untuk mendapatkan item dari tabel Amazon DynamoDB bernama.

MyCompany Anda juga dapat mengizinkan John untuk mengelola kredensyal keamanan IAM-nya sendiri. Kebijakan berbasis identitas dapat terkelola atau inline.

Kebijakan berbasis sumber daya dilampirkan pada sumber daya. Misalnya, Anda dapat melampirkan kebijakan berbasis sumber daya ke bucket Amazon S3, antrian Amazon SQS, titik akhir VPC, kunci enkripsi, serta tabel dan aliran Amazon DynamoDB. AWS Key Management Service Untuk daftar layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM.

Dengan kebijakan berbasis sumber daya, Anda dapat menentukan siapa yang memiliki akses ke sumber daya tersebut dan tindakan apa yang dapat mereka lakukan di situ. Untuk mempelajari apakah prinsipal dalam akun di luar zona kepercayaan (organisasi atau akun terpercaya) memiliki akses untuk mengambil peran Anda, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?. Kebijakan berbasis sumber daya hanya bersifat inline, tidak terkelola.

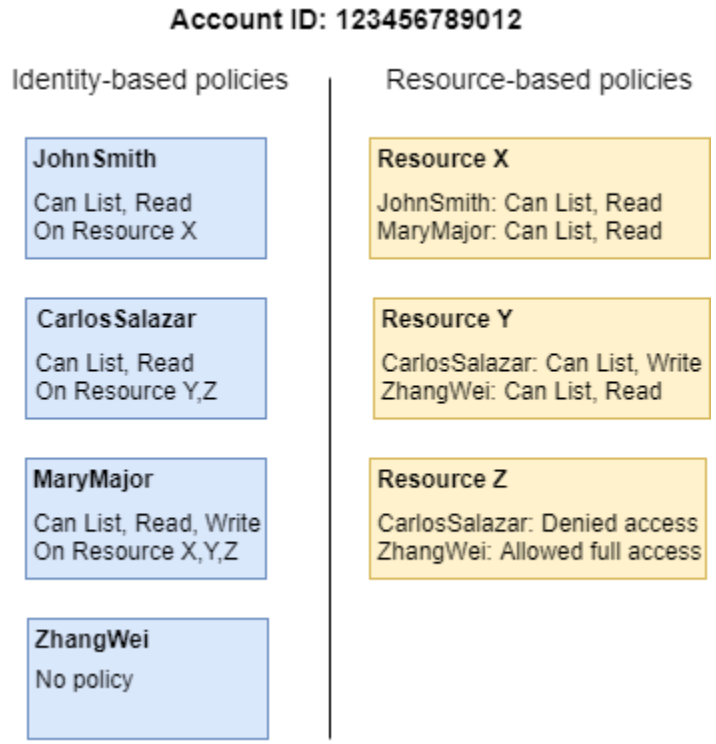
> **Note**
>
> Kebijakan Berbasis sumber daya berbeda dari izin tingkat sumber daya. Anda dapat melampirkan kebijakan berbasis sumber daya secara langsung ke sumber daya, sebagaimana dijelaskan dalam topik ini. Izin tingkat sumber daya mengacu pada kemampuan untuk menggunakan ARN untuk menentukan sumber daya individu dalam kebijakan. Kebijakan berbasis sumber daya hanya didukung oleh beberapa layanan. AWS Untuk daftar yang layanannya didukung kebijakan berbasis sumber daya dan izin tingkat sumber daya, lihat AWS layanan yang bekerja dengan IAM.

Untuk mempelajari cara berinteraksi antara kebijakan berbasis identitas dan kebijakan berbasis sumber daya dalam akun yang sama, lihat Mengevaluasi kebijakan dalam satu akun.

Untuk mempelajari cara interaksi kebijakan di seluruh akun, lihat Logika evaluasi kebijakan lintas akun.

Untuk lebih memahami konsep ini, lihat gambar berikut. Administrator dari akun `123456789012` terlampir kebijakan berbasis identitas kepada pengguna `JohnSmith`, `CarlosSalazar`, dan `MaryMajor`. Beberapa tindakan dalam kebijakan ini dapat dilakukan pada sumber daya tertentu. Misalnya, pengguna `JohnSmith` dapat melakukan beberapa tindakan di `Resource X`. Ini adalah izin tingkat sumber daya dalam kebijakan berbasis identitas. Administrator juga menambahkan kebijakan berbasis sumber daya ke `Resource X`, `Resource Y`, dan `Resource Z`. Kebijakan berbasis sumber daya memungkinkan Anda menentukan siapa yang dapat mengakses sumber daya

tersebut. Misalnya, kebijakan berbasis sumber daya di `Resource X` mengizinkan akses pengguna `JohnSmith` dan `MaryMajor` daftar dan baca ke sumber daya.



Contoh akun `123456789012` mengizinkan pengguna berikut untuk melakukan tindakan yang tercantum:

- JohnSmith— John dapat melakukan daftar dan membaca tindakan di`Resource X`. Dia diberikan izin ini oleh kebijakan berbasis identitas pada penggunanya dan kebijakan berbasis sumber daya pada `Resource X`.
- CarlosSalazarCarlos dapat melakukan daftar, membaca, dan menulis tindakan`Resource Y`, tetapi ditolak aksesnya. `Resource Z` Kebijakan berbasis identitas pada Carlos memungkinkan dia untuk melakukan tindakan daftar dan baca `Resource Y`. Kebijakan berbasis sumber daya `Resource Y` juga memungkinkan dia menulis izin. Namun, meskipun kebijakan berbasis identitas miliknya mengizinkannya mengakses `Resource Z`, kebijakan berbasis sumber daya `Resource Z` menolak akses tersebut. `Deny` secara jelas menimpa `Allow` dan aksesnya ke `Resource Z` ditolak. Untuk informasi selengkapnya, lihat Logika evaluasi kebijakan.
- MaryMajor— Mary dapat melakukan operasi daftar, membaca, dan menulis pada`Resource X`,`Resource Y`, dan`Resource Z`. Kebijakan berbasis identitas miliknya mengizinkan tindakan yang lebih banyak terhadap lebih banyak sumber daya daripada kebijakan berbasis sumber daya, tetapi tidak ada yang menolak akses.

- ZhangWei— Zhang memiliki akses penuh ke`Resource Z`. Zhang tidak memiliki kebijakan berbasis identitas, tetapi kebijakan berbasis sumber daya `Resource Z` mengizinkannya mengakses sumber daya sepenuhnya. Zhang juga dapat melakukan tindakan daftar dan baca pada `Resource Y`.

Kebijakan berbasis identitas dan kebijakan berbasis sumber daya keduanya adalah kebijakan izin dan dievaluasi bersama. Untuk permintaan yang hanya berlaku kebijakan izin, AWS pertama-tama periksa semua kebijakan untuk file. `Deny` Jika ada, maka permintaan ditolak. Kemudian AWS memeriksa setiap `Allow`. Jika setidaknya satu pernyataan mengizinkan tindakan dalam permintaan tersebut, permintaan tersebut diizinkan. Tidak peduli apakah `Allow` berada dalam kebijakan berbasis identitas atau kebijakan berbasis sumber daya.

> **Important**
>
> Logika ini hanya berlaku ketika permintaan dibuat dalam satu Akun AWS. Untuk permintaan yang dibuat dari satu akun ke akun lain, pemohon di `Account A` harus memiliki kebijakan berbasis identitas yang mengizinkan mereka mengajukan permintaan kepada sumber daya di `Account B`. Juga, kebijakan berbasis sumber daya di `Account B` harus mengizinkan pemohon di `Account A` untuk mengakses sumber daya. Harus ada kebijakan di kedua akun yang mengizinkan operasi, jika tidak permintaan tersebut gagal. Untuk informasi selengkapnya tentang menggunakan kebijakan berbasis sumber daya bagi akses akun silang, lihat Akses sumber daya lintas akun di IAM.

Pengguna yang memiliki izin spesifik mungkin meminta sumber daya yang juga memiliki kebijakan izin yang terlampir. Dalam hal ini, AWS mengevaluasi kedua set izin saat menentukan apakah akan memberikan akses ke sumber daya. Untuk informasi tentang bagaimana kebijakan dievaluasi, lihat Logika evaluasi kebijakan.

> **Note**
>
> Amazon S3 mendukung kebijakan berbasis identitas dan kebijakan berbasis sumber daya (disebut sebagai kebijakan bucket). Sebagai tambahan, Amazon S3 mendukung mekanisme izin yang dikenal sebagai Access Control List (ACL) yang terpisah dari kebijakan dan izin IAM. Anda dapat menggunakan kebijakan IAM dalam kombinasi dengan Amazon S3

ACL. Untuk informasi selengkapnya, lihat Kontrol Akses di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Kontrol akses ke AWS sumber daya menggunakan kebijakan

Anda dapat menggunakan kebijakan untuk mengontrol akses ke sumber daya di dalam IAM atau semua sumber daya AWS.

Untuk menggunakan kebijakan untuk mengontrol akses AWS, Anda harus memahami cara AWS memberikan akses. AWS terdiri dari koleksi sumber daya. Pengguna IAM adalah sebuah sumber daya. Bucket Amazon S3 adalah sebuah sumber daya. Ketika Anda menggunakan AWS API, the AWS CLI, atau AWS Management Console untuk melakukan operasi (seperti membuat pengguna), Anda mengirim permintaan untuk operasi itu. Permintaan Anda menentukan tindakan, sumber daya, sebuah entitas prinsipal (pengguna atau peran), sebuah akun prinsipal, dan informasi permintaan yang diperlukan. Semua informasi ini memberikan konteks.

AWS kemudian memeriksa bahwa Anda (kepala sekolah) diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk melakukan tindakan yang ditentukan pada sumber daya yang ditentukan. Selama otorisasi, AWS periksa semua kebijakan yang berlaku untuk konteks permintaan Anda. Sebagian besar kebijakan disimpan AWS sebagai JSONdokumen dan menentukan izin untuk entitas utama. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

AWS mengotorisasi permintaan hanya jika setiap bagian dari permintaan Anda diizinkan oleh kebijakan. Untuk melihat diagram proses ini, lihat Cara kerja IAM. Untuk detail tentang cara AWS menentukan apakah permintaan diizinkan, lihatLogika evaluasi kebijakan.

Saat membuat IAM kebijakan, Anda dapat mengontrol akses ke hal-hal berikut:

- Prinsipal – Kontrol apa yang boleh dilakukan oleh orang yang membuat permintaan (prinsipal).
- IAMIdentitas — Kontrol IAM identitas (IAMgrup, pengguna, dan peran) mana yang dapat diakses dan bagaimana caranya.
- IAMKebijakan — Kontrol siapa yang dapat membuat, mengedit, dan menghapus kebijakan yang dikelola pelanggan, dan siapa yang dapat melampirkan dan melepaskan semua kebijakan yang dikelola.
- AWS Sumber Daya — Kontrol siapa yang memiliki akses ke sumber daya menggunakan kebijakan berbasis identitas atau kebijakan berbasis sumber daya.

- AWS Akun — Kontrol apakah permintaan hanya diperbolehkan untuk anggota akun tertentu.

Kebijakan memungkinkan Anda menentukan siapa yang memiliki akses ke AWS sumber daya, dan tindakan apa yang dapat mereka lakukan pada sumber daya tersebut. Setiap IAM pengguna memulai tanpa izin. Dengan kata lain, secara default, pengguna tidak dapat melakukan apa pun, bahkan tidak dapat melihat access key milik mereka. Untuk memberikan izin kepada pengguna untuk melakukan sesuatu, Anda dapat menambahkan izin kepada pengguna (yaitu, melampirkan kebijakan ke pengguna). Atau, Anda dapat menambahkan pengguna ke grup pengguna yang memiliki izin yang dimaksud.

Misalnya, Anda dapat memberikan izin pengguna untuk mencantumkan access key-nya sendiri. Anda juga dapat memperluas izin tersebut dan mengizinkan setiap pengguna membuat, memperbarui, serta menghapus kunci milik mereka.

Saat Anda memberikan izin kepada grup pengguna, semua pengguna dalam grup pengguna tersebut mendapatkan izin tersebut. Misalnya, Anda dapat memberikan izin kepada grup pengguna Administrator untuk melakukan IAM tindakan apa pun pada Akun AWS sumber daya apa pun. Contoh lain: Anda dapat memberikan izin kepada grup pengguna Manajer untuk menjelaskan EC2 instans Amazon. Akun AWS

Untuk informasi tentang cara mendelegasikan izin dasar kepada pengguna, IAM grup, dan peran Anda, lihat. Izin diperlukan untuk mengakses sumber daya IAM Untuk contoh tambahan dari kebijakan yang menggambarkan izin dasar, lihat Contoh kebijakan untuk mengelola sumber daya IAM.

Mengontrol akses untuk prinsipal

Anda dapat menggunakan kebijakan untuk mengontrol apa yang boleh dilakukan oleh orang yang membuat permintaan (prinsipal). Untuk melakukannya, Anda harus melampirkan kebijakan berbasis identitas pada identitas prang tersebut (pengguna, grup pengguna, atau peran). Anda juga dapat menggunakan batas izin untuk mengatur izin maksimum yang dapat dimiliki sebuah entitas (pengguna atau peran).

Misalnya, asumsikan bahwa Anda ingin pengguna Zhang Wei memiliki akses penuh ke, Amazon DynamoDB CloudWatch, AmazonEC2, dan Amazon S3. Anda dapat membuat dua kebijakan berbeda agar kemudian Anda dapat memisahkannya jika Anda memerlukan satu set izin untuk pengguna yang berbeda. Atau Anda dapat menempatkan kedua izin bersama-sama dalam satu kebijakan, dan kemudian melampirkan kebijakan itu ke IAM pengguna yang bernama Zhang Wei. Anda juga dapat melampirkan kebijakan ke grup pengguna di mana Zhang berada, atau peran

yang dapat diambil oleh Zhang. Hasilnya, saat Zhang melihat konten di bucket S3, permintaannya diizinkan. Jika dia mencoba membuat IAM pengguna baru, permintaannya ditolak karena dia tidak memiliki izin.

Anda dapat menggunakan batas izin pada Zhang untuk memastikan bahwa dia tidak pernah diberi akses ke bucket S3 `amzn-s3-demo-bucket1`. Untuk melakukannya, tentukan maksimum izin yang Anda inginkan Zhang untuk memilikinya. Dalam kasus ini, Anda mengontrol apa yang dia lakukan dengan menggunakan kebijakan izinnya. Di sini, Anda hanya peduli bahwa dia tidak mengakses bucket rahasia. Jadi, Anda menggunakan kebijakan berikut untuk menentukan batas Zhang untuk mengizinkan semua AWS tindakan untuk Amazon S3 dan beberapa layanan lainnya tetapi menolak akses ke bucket S3. `amzn-s3-demo-bucket1` Karena batas izin tidak mengizinkan IAM tindakan apa pun, itu mencegah Zhang menghapus batasnya (atau siapa pun).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
                "ec2:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1",
                "arn:aws:s3:::amzn-s3-demo-bucket1/*"
            ]
        }
    ]
}
```

Saat Anda menetapkan kebijakan seperti ini sebagai batas izin bagi pengguna, ingatlah bahwa itu tidak memberikan izin apa pun. Ini menetapkan izin maksimum yang dapat diberikan oleh kebijakan

berbasis identitas kepada entitas. IAM Untuk informasi lebih lanjut tentang batas izin, lihat Batas izin untuk entitas IAM.

Untuk informasi terperinci tentang prosedur yang disebutkan sebelumnya, lihat sumber daya ini:

- Untuk mempelajari lebih lanjut tentang membuat IAM kebijakan yang dapat Anda lampirkan ke kepala sekolah, lihatTentukan IAM izin khusus dengan kebijakan terkelola pelanggan.
- Untuk mempelajari cara melampirkan IAM kebijakan ke kepala sekolah, lihatMenambahkan dan menghapus izin identitas IAM.
- Untuk melihat contoh kebijakan untuk memberikan akses penuh keEC2, lihatAmazon EC2: Memungkinkan akses EC2 penuh dalam Wilayah tertentu, secara terprogram dan di konsol.
- Untuk memungkinkan akses hanya-baca ke bucket S3, gunakan dua pernyataan pertama dari contoh kebijakan berikut: Amazon S3: Memungkinkan akses baca dan tulis ke objek di Bucket S3, secara terprogram dan di konsol.
- Untuk melihat contoh kebijakan yang memungkinkan pengguna menyetel kredensialnya, seperti kata sandi konsol, kunci akses terprogram, dan MFA perangkat mereka, lihat. AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan

Mengontrol akses ke identitas

Anda dapat menggunakan IAM kebijakan untuk mengontrol apa yang dapat dilakukan pengguna terhadap identitas dengan membuat kebijakan yang Anda lampirkan ke semua pengguna melalui grup pengguna. Untuk melakukan ini, buat kebijakan yang membatasi apa yang dapat dilakukan pada sebuah identitas, atau siapa yang dapat mengaksesnya.

Misalnya, Anda dapat membuat grup pengguna bernama AllUsers, lalu melampirkan grup pengguna tersebut ke semua pengguna. Saat membuat grup pengguna, Anda dapat memberikan akses kepada semua pengguna untuk mengatur kredensialnya seperti yang dijelaskan di bagian sebelumnya. Anda kemudian dapat membuat kebijakan yang menolak akses untuk mengubah grup pengguna kecuali jika nama pengguna disertakan dalam ketentuan dari kebijakan tersebut. Namun bagian dari kebijakan tersebut hanya akan menolak akses ke siapa pun kecuali pengguna yang tercantum. Anda juga harus menyertakan izin yang mengizinkan semua tindakan manajemen grup pengguna untuk semua orang di dalam grup pengguna. Terakhir, Anda melampirkan kebijakan ini ke grup pengguna sehingga itu diterapkan ke semua pengguna. Hasilnya, saat pengguna yang tidak disebutkan dalam kebijakan mencoba melakukan perubahan pada grup pengguna, permintaannya ditolak.

Untuk membuat kebijakan ini dengan editor visual

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi di sebelah kiri, pilih Kebijakan.

   Jika ini pertama kalinya Anda memilih Kebijakan, akan muncul halaman Selamat Datang di Kebijakan Terkelola. Pilih Memulai.
3. Pilih Buat kebijakan.
4. Pada bagian Editor kebijakan, pilih opsi Visual.
5. Di Pilih layanan pilih IAM.
6. Di Tindakan yang diizinkan, **`group`** ketik kotak pencarian. Editor visual menunjukkan semua tindakan IAM yang berisi kata `group`. Pilih semua kotak centang.
7. Pilih Sumber Daya untuk menentukan sumber daya bagi kebijakan Anda. Berdasarkan tindakan yang Anda pilih, Anda akan melihat jenis sumber daya grup dan pengguna.

   - grup - Pilih Tambah ARNs. Untuk Sumber Daya di, pilih opsi Akun apa pun. Pilih kotak centang Nama grup apa pun dengan jalur lalu ketik nama grup pengguna**`AllUsers`**. Kemudian pilih TambahkanARNs.
   - pengguna — Pilih kotak centang di sebelah Apa saja di akun ini.

   Salah satu tindakan yang Anda pilih, `ListGroups`, tidak mendukung penggunaan sumber daya tertentu. Anda tidak harus memilih Semua sumber daya untuk tindakan itu. Saat menyimpan kebijakan atau melihat kebijakan di JSONeditor, Anda dapat melihat bahwa IAM secara otomatis membuat blok izin baru yang memberikan izin tindakan ini pada semua sumber daya.
8. Untuk menambahkan blok izin lain, pilih Tambahkan izin lainnya.
9. Pilih Pilih layanan dan kemudian pilih IAM.
10. Pilih Tindakan yang diizinkan dan kemudian pilih Beralih untuk menolak izin. Saat Anda melakukannya, seluruh blok digunakan untuk menolak izin.
11. Jenis **`group`** di kotak pencarian. Editor visual menunjukkan semuatindakan IAM yang berisi kata `group`. Pilih kotak centang di sebelah tindakan berikut:

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup

- AttachGroupPolicy
- DeleteGroupPolicy
- DetachGroupPolicy
- PutGroupPolicy
- UpdateGroup

12. Pilih Sumber Daya untuk menentukan sumber daya bagi kebijakan Anda. Berdasarkan tindakan yang Anda pilih, Anda akan melihat jenis sumber daya grup. Pilih Tambah ARNs. Untuk Sumber Daya di, pilih opsi Akun apa pun. Untuk Setiap nama grup dengan jalur, ketik nama grup pengguna**`AllUsers`**. Kemudian pilih TambahkanARNs.
13. Pilih kondisi Permintaan - opsional dan kemudian pilih Tambahkan kondisi lain. Lengkapi formulir dengan nilai berikut:

    - Kunci kondisi - Pilih aws:username
    - Pengukur – Pilih Default
    - Operator – Pilih StringNotEquals
    - Nilai — Ketik **`srodriguez`** dan kemudian pilih Tambah untuk menambahkan nilai lain. Ketik **`mjackson`** dan kemudian pilih Tambah untuk menambahkan nilai lain. Ketik **`adesai`** dan kemudian pilih Tambahkan kondisi.

    Ketentuan ini memastikan bahwa akses akan ditolak untuk tindakan manajemen grup pengguna tertentu saat pengguna membuat panggilan tidak dimasukkan dalam daftar. Karena ini secara jelas menolak izin, ini menggantikan blok sebelumnya yang mengizinkan pengguna tersebut untuk memanggil tindakan. Pengguna dalam daftar tidak ditolak aksesnya, dan mereka diberikan izin pada blok izin pertama, sehingga mereka dapat sepenuhnya mengelola grup pengguna tersebut.
14. Setelah selesai, pilih Selanjutnya.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di opsi Editor visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

15. Pada halaman Tinjau dan buat, untuk Nama Kebijakan, ketik**`LimitAllUserGroupManagement`**. Untuk bagian Description (Deskripsi), ketik **`Allows all users read-only access to a specific user group, and allows only specific users access to make changes to the user group`**. Tinjau Izin yang ditentukan dalam kebijakan ini untuk memastikan bahwa Anda telah memberikan izin yang dimaksud. Kemudian pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.
16. Lampirkan kebijakan tersebut ke grup pengguna Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Atau, Anda dapat membuat kebijakan yang sama menggunakan contoh dokumen JSON kebijakan ini. Untuk melihat JSON kebijakan ini, lihatIAM: Memungkinkan pengguna IAM tertentu untuk mengelola grup secara terprogram dan di konsol. Untuk petunjuk terperinci untuk membuat kebijakan menggunakan JSON dokumen, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

Mengendalikan akses ke kebijakan

Anda dapat mengontrol cara pengguna menerapkan kebijakan AWS terkelola. Untuk melakukannya, lampirkan kebijakan ini ke semua pengguna Anda. Idealnya, Anda dapat melakukan ini menggunakan sebuah grup pengguna.

Misalnya, Anda dapat membuat kebijakan yang memungkinkan pengguna hanya melampirkan kebijakan IAMUserChangePassworddan PowerUserAccess AWS terkelola ke IAM pengguna, grup pengguna, atau peran baru.

Untuk kebijakan yang dikelola pelanggan, Anda dapat mengontrol siapa yang dapat membuat, memperbarui, dan menghapus kebijakan ini. Anda dapat mengontrol siapa yang dapat melampirkan dan melepaskan kebijakan ke dan dari entitas utama (IAMgrup, pengguna, dan peran). Anda juga dapat mengontrol kebijakan mana yang dapat diberikan atau dilepas pengguna, dan ke dan dari entitas mana.

Misalnya, Anda dapat memberikan izin kepada administrator akun untuk membuat, memperbarui, dan menghapus kebijakan. Kemudian Anda memberikan izin kepada pemimpin tim atau administrator terbatas lainnya untuk memberikan dan melepaskan kebijakan ini ke dan dari entitas prinsipal yang dikelola oleh administrator terbatas.

Untuk informasi selengkapnya, lihat sumber daya ini:

- Untuk mempelajari lebih lanjut tentang membuat IAM kebijakan yang dapat Anda lampirkan ke kepala sekolah, lihatTentukan IAM izin khusus dengan kebijakan terkelola pelanggan.
- Untuk mempelajari cara melampirkan IAM kebijakan ke kepala sekolah, lihatMenambahkan dan menghapus izin identitas IAM.
- Untuk melihat contoh kebijakan untuk membatasi penggunaan kebijakan yang dikelola, lihat IAM: Membatasi kebijakan terkelola yang dapat diterapkan pada pengguna, grup, atau peran IAM.

Mengontrol izin untuk membuat, memperbarui, dan menghapus kebijakan yang dikelola pelanggan

Anda dapat menggunakan IAMkebijakan untuk mengontrol siapa yang diizinkan untuk membuat, memperbarui, dan menghapus kebijakan yang dikelola pelanggan di Anda Akun AWS. Daftar berikut berisi API operasi yang berkaitan langsung dengan pembuatan, pembaruan, dan penghapusan kebijakan atau versi kebijakan:

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

APIOperasi dalam daftar sebelumnya sesuai dengan tindakan yang dapat Anda izinkan atau tolak—yaitu izin yang dapat Anda berikan — menggunakan kebijakan. IAM

Pertimbangkan contoh kebijakan berikut. Ini memungkinkan pengguna untuk membuat, memperbarui (yaitu, membuat versi kebijakan baru), menghapus, dan menetapkan versi default untuk semua kebijakan yang dikelola pelanggan di Akun AWS. Contoh kebijakan juga mengizinkan pengguna untuk mencantumkan kebijakan dan mendapatkan kebijakan. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

Example Contoh kebijakan yang mengizinkan membuat, memperbarui, menghapus, mencantumkan, mendapatkan, dan mengatur versi default untuk semua kebijakan

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
      "iam:DeletePolicy",
      "iam:DeletePolicyVersion",
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "*"
  }
}
```

Anda dapat membuat kebijakan yang membatasi penggunaan API operasi ini agar hanya memengaruhi kebijakan terkelola yang Anda tentukan. Misalnya, Anda mungkin ingin mengizinkan pengguna untuk mengatur versi default dan menghapus versi kebijakan, tetapi hanya untuk kebijakan tertentu yang dikelola pelanggan. Anda melakukannya dengan menentukan kebijakan ARN dalam `Resource` elemen kebijakan yang memberikan izin ini.

Contoh berikut menunjukkan kebijakan yang mengizinkan pengguna menghapus versi kebijakan dan mengatur versi default. Tetapi tindakan ini hanya diperbolehkan untuk kebijakan yang dikelola pelanggan yang menyertakan jalur/TEAM-A/. Kebijakan yang dikelola pelanggan ARN ditentukan dalam `Resource` elemen kebijakan. (Dalam contoh ini ARN menyertakan jalur dan wildcard dan dengan demikian cocok dengan semua kebijakan yang dikelola pelanggan yang menyertakan jalur/TEAM-A/). Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

Untuk informasi selengkapnya tentang menggunakan jalur atas nama kebijakan yang dikelola pelanggan, lihat Nama dan jalur yang ramah.

Example Contoh kebijakan yang mengizinkan menghapus versi kebijakan dan mengatur versi default hanya untuk kebijakan tertentu

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:DeletePolicyVersion",
```

```
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "arn:aws:iam::account-id:policy/TEAM-A/*"
  }
}
```

Mengontrol izin untuk memberikan dan melepaskan kebijakan yang dikelola

Anda juga dapat menggunakan IAM kebijakan untuk memungkinkan pengguna bekerja hanya dengan kebijakan terkelola tertentu. Sebagai akibatnya, Anda dapat mengontrol izin mana yang diizinkan untuk diberikan pengguna kepada entitas prinsipal lainnya.

Daftar berikut menunjukkan API operasi yang berkaitan langsung dengan melampirkan dan melepaskan kebijakan terkelola ke dan dari entitas utama:

- AttachGroupPolicy
- AttachRolePolicy
- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

Anda dapat membuat kebijakan yang membatasi penggunaan API operasi ini untuk hanya memengaruhi kebijakan terkelola tertentu dan/atau entitas utama yang Anda tentukan. Misalnya, Anda mungkin ingin mengizinkan pengguna untuk melampirkan kebijakan terkelola, tetapi hanya kebijakan terkelola yang Anda tentukan. Atau, Anda mungkin ingin mengizinkan pengguna untuk memberikan kebijakan terkelola, tetapi hanya kepada entitas prinsipal yang Anda tentukan.

Contoh kebijakan berikut memungkinkan pengguna untuk melampirkan kebijakan terkelola hanya ke IAM grup dan peran yang menyertakan jalur/TEAM-A/. Grup pengguna dan peran ARNs ditentukan dalam `Resource` elemen kebijakan. (Dalam contoh ini ARNs sertakan jalur dan karakter wildcard dan dengan demikian cocok dengan semua IAM grup dan peran yang menyertakan jalur/TEAM-A/). Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

Example Contoh kebijakan yang mengizinkan memberikan kebijakan terkelola hanya untuk grup pengguna atau peran tertentu

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ]
  }
}
```

Anda selanjutnya dapat membatasi tindakan dalam contoh sebelumnya untuk hanya memengaruhi kebijakan tertentu. Artinya, Anda dapat mengontrol izin yang boleh diberikan pengguna ke entitas prinsipal lainnya—dengan menambahkan ketentuan ke kebijakan.

Dalam contoh berikut, ketentuan ini memastikan bahwa izin `AttachGroupPolicy` dan `AttachRolePolicy` hanya diperbolehkan jika kebijakan yang diberikan sesuai dengan salah satu kebijakan yang ditentukan. Ketentuan menggunakan `iam:PolicyARN` kunci ketentuan untuk menentukan kebijakan mana yang diizinkan untuk diberikan. Contoh kebijakan berikut berkembang pada contoh sebelumnya. Ini memungkinkan pengguna untuk melampirkan hanya kebijakan terkelola yang menyertakan jalur/TEAM-A/ ke hanya IAM grup dan peran yang menyertakan jalur/-A/TEAM. Untuk mempelajari cara membuat kebijakan menggunakan contoh dokumen JSON kebijakan ini, lihatthe section called “Membuat kebijakan menggunakan JSON editor”.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
```

```
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ],
    "Condition": {"ArnLike":
      {"iam:PolicyARN": "arn:aws:iam::account-id:policy/TEAM-A/*"}
    }
  }
}
```

Kebijakan ini menggunakan operator `ArnLike` kondisi, tetapi Anda juga dapat menggunakan operator `ArnEquals` kondisi karena kedua operator kondisi ini berperilaku identik. Untuk informasi lebih lanjut tentang `ArnLike` dan `ArnEquals`, lihat Operator kondisi Nama Sumber Daya Amazon (ARN) dalam Jenis Ketentuan bagian dari Referensi Elemen Kebijakan.

Misalnya, Anda dapat membatasi penggunaan tindakan untuk hanya melibatkan kebijakan terkelola yang Anda tentukan. Anda melakukannya dengan menentukan kebijakan ARN dalam `Condition` elemen kebijakan yang memberikan izin ini. Misalnya, untuk menentukan kebijakan ARN yang dikelola pelanggan:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

Anda juga dapat menentukan ARN kebijakan AWS terkelola dalam `Condition` elemen kebijakan. Kebijakan ARN AWS terkelola menggunakan alias khusus `aws` dalam kebijakan, ARN bukan ID akun, seperti dalam contoh ini:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Mengontrol akses ke sumber daya

Anda dapat mengontrol akses ke sumber daya dengan menggunakan kebijakan berbasis sumber daya. Dalam kebijakan berbasis identitas, Anda melampirkan kebijakan ke identitas dan menyebutkan sumber daya apa yang dapat diakses oleh identitas tersebut. Dalam kebijakan berbasis sumber daya, Anda memberikan kebijakan pada sumber daya yang ingin Anda kontrol. Dalam kebijakan, Anda menentukan prinsipal mana yang dapat mengakses sumber daya tersebut. Untuk informasi lebih lanjut tentang kedua jenis kebijakan, lihat Kebijakan berbasis identitas dan kebijakan berbasis sumber daya.

Untuk informasi selengkapnya, lihat sumber daya ini:

- Untuk mempelajari lebih lanjut tentang membuat IAM kebijakan yang dapat Anda lampirkan ke kepala sekolah, lihatTentukan IAM izin khusus dengan kebijakan terkelola pelanggan.
- Untuk mempelajari cara melampirkan IAM kebijakan ke kepala sekolah, lihatMenambahkan dan menghapus izin identitas IAM.
- Amazon S3 mendukung penggunaan kebijakan berbasis sumber daya di bucket mereka. Untuk informasi lebih lanjut, lihat Contoh Kebijakan Bucket:

Pembuat Sumber Daya Tidak Secara Otomatis Memiliki Izin

Jika Anda masuk menggunakan Pengguna root akun AWS kredensi, Anda memiliki izin untuk melakukan tindakan apa pun pada sumber daya milik akun. Namun, ini tidak berlaku untuk IAM pengguna. IAMPengguna mungkin diberikan akses untuk membuat sumber daya, tetapi izin pengguna, bahkan untuk sumber daya itu, terbatas pada apa yang telah diberikan secara eksplisit. Ini berarti bahwa hanya karena Anda membuat sumber daya, seperti IAM peran, Anda tidak secara otomatis memiliki izin untuk mengedit atau menghapus peran itu. Sebagai tambahan, izin Anda dapat dicabut setiap saat oleh pemilik akun atau pengguna lain yang telah diberi akses untuk mengelola izin Anda.

Mengontrol akses ke prinsipal dalam akun tertentu

Anda dapat langsung memberikan IAM pengguna di akun Anda sendiri akses ke sumber daya Anda. Jika pengguna dari akun lain memerlukan akses ke sumber daya Anda, Anda dapat membuat IAM peran. Peran adalah sebuah entitas yang mencakup izin tetapi tidak terkait dengan pengguna tertentu. Pengguna dari akun lain kemudian dapat menggunakan peran dan mengakses sumber daya sesuai dengan izin yang telah Anda tetapkan untuk peran tersebut. Untuk informasi selengkapnya, lihat Akses untuk IAM pengguna lain Akun AWS yang Anda miliki.

> **Note**
>
> Beberapa layanan mendukung kebijakan berbasis sumber daya seperti yang dijelaskan dalam Kebijakan berbasis identitas dan kebijakan berbasis sumber daya (seperti Amazon S3, Amazon, dan Amazon). SNS SQS Untuk layanan tersebut, alternatif untuk menggunakan peran adalah dengan memberikan kebijakan ke sumber daya (bucket, topik, atau antrean) yang ingin Anda bagikan. Kebijakan berbasis sumber daya dapat menentukan AWS akun yang memiliki izin untuk mengakses sumber daya.

Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag

Gunakan informasi di bagian berikut untuk mengontrol siapa yang dapat mengakses pengguna dan peran IAM Anda serta sumber daya apa yang dapat diakses pengguna dan peran Anda. Untuk informasi lebih umum dan contoh pengendalian akses ke AWS sumber daya lain, termasuk sumber daya IAM lainnya, lihatTag untuk AWS Identity and Access Management sumber daya.

> **Note**
>
> Untuk detail tentang sensitivitas huruf besar untuk kunci tag dan nilai kunci tag, lihatCase sensitivity.

Tag dapat diberikan ke sumber daya IAM, melewati permintaan, atau diberikan pada prinsipal yang membuat permintaan. Pengguna atau peran IAM dapat berupa sumber daya dan prinsipal. Misalnya, Anda dapat menulis sebuah kebijakan yang mengizinkan pengguna untuk mencantumkan grup untuk pengguna. Operasi ini hanya diizinkan jika pengguna yang mengajukan permintaan (principal) memiliki tanda `project=blue` yang sama sebagai pengguna yang mereka coba lihat. Dalam contoh ini, pengguna dapat melihat keanggotaan kelompok untuk pengguna mana pun, termasuk mereka sendiri, selama mereka bekerja pada proyek yang sama.

tuanUntuk mengontrol akses berdasarkan tag, Anda memberikan informasi tag di elemen ketentuan dari kebijakan. Saat Anda membuat kebijakan IAM, Anda dapat menggunakan tag IAM dan kunci ketentuan tag terkait untuk mengontrol akses ke salah satu hal berikut:

- Sumber Daya – Mengontrol akses ke sumber daya pengguna atau peran berdasarkan tag mereka. Untuk melakukan ini, gunakan kunci kondisi aws:ResourceTag/***key-name*** untuk menentukan pasangan nilai kunci tag mana yang harus dilampirkan ke sumber daya. Untuk informasi selengkapnya, lihat Mengontrol akses ke sumber daya AWS.
- Permintaan – Mengontrol tag yang dapat diberikan dalam permintaan IAM. Untuk melakukan ini, gunakan kunci kondisi ***nama kunci*** aws:RequestTag/untuk menentukan tag apa yang dapat ditambahkan, diubah, atau dihapus dari pengguna atau peran IAM. Kunci ini digunakan dengan cara yang sama untuk sumber daya IAM dan sumber AWS daya lainnya. Untuk informasi selengkapnya, lihat Mengontrol akses selama permintaan AWS.
- Prinsipal – Mengontrol apa yang boleh dilakukan oleh orang yang membuat permintaan (prinsipal) berdasarkan tag yang dilampirkan ke pengguna atau peran IAM orang tersebut. Untuk melakukan ini, gunakan kunci kondisi ***nama kunci*** aws:PrincipalTag/untuk menentukan tag apa yang harus dilampirkan ke pengguna atau peran IAM sebelum permintaan diizinkan.

- Setiap bagian dari proses otorisasi — Gunakan aws: TagKeys condition key untuk mengontrol apakah kunci tag tertentu dapat digunakan dalam permintaan atau oleh prinsipal. Dalam hal ini, nilai kunci tidak menjadi masalah. Kunci ini berperilaku sama untuk IAM dan layanan lainnya AWS . Namun, ketika Anda menandai pengguna di IAM, ini juga mengontrol apakah prinsipal dapat membuat permintaan ke layanan apa pun. Untuk informasi selengkapnya, lihat Mengontrol akses berdasarkan kunci tanda.

Anda dapat membuat kebijakan IAM menggunakan editor visual, dengan menggunakan JSON, atau dengan mengimpor kebijakan terkelola yang ada. Untuk detailnya, lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan.

> **Note**
>
> Anda juga dapat meneruskan tag sesi saat Anda mengambil peran IAM atau mengfederasi pengguna. Ini hanya berlaku untuk durasi sesi.

Mengontrol akses untuk prinsipal IAM

Anda dapat mengontrol apa yang diperbolehkan untuk penanggung jawab berdasarkan tanda yang terpasang pada identitas orang tersebut.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan setiap pengguna di akun ini untuk melihat keanggotaan grup untuk setiap pengguna, termasuk mereka sendiri, selama mereka mengerjakan proyek yang sama. Operasi ini hanya diperbolehkan jika tag sumber daya pengguna dan tag prinsipal memiliki nilai yang sama untuk kunci tag`project`. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
            "Condition": {
```

```
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
            }
        }]
}
```

Mengontrol akses berdasarkan kunci tag

Anda dapat menggunakan tag dalam kebijakan IAM Anda untuk mengontrol apakah kunci tag tertentu dapat digunakan dalam permintaan atau oleh prinsipal.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan hanya menghapus tag dengan `temporary` kunci dari pengguna. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": ["temporary"]}}
    }]
}
```

Mengontrol akses ke AWS sumber daya menggunakan tag

Anda dapat menggunakan tag untuk mengontrol akses ke AWS sumber daya yang mendukung penandaan, termasuk IAM sumber daya. Anda dapat menandai IAM pengguna dan peran untuk mengontrol apa yang dapat mereka akses. Untuk mempelajari cara menandai IAM pengguna dan peran, lihatTag untuk AWS Identity and Access Management sumber daya. Selain itu, Anda dapat mengontrol akses ke IAM sumber daya berikut: kebijakan yang dikelola pelanggan, penyedia IAM identitas, profil instans, sertifikat server, dan MFA perangkat virtual. Untuk melihat tutorial untuk membuat dan menguji kebijakan yang memungkinkan IAM peran dengan tag utama untuk mengakses sumber daya dengan tag yang cocok, lihatIAMtutorial: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag. Gunakan informasi di bagian berikut untuk mengontrol akses ke AWS sumber daya lain, termasuk IAM sumber daya, tanpa menandai IAM pengguna atau peran.

Sebelum Anda menggunakan tag untuk mengontrol akses ke AWS sumber daya Anda, Anda harus memahami bagaimana AWS memberikan akses. AWS terdiri dari koleksi sumber daya. EC2Instans Amazon adalah sumber daya. Bucket Amazon S3 adalah sebuah sumber daya. Anda dapat menggunakan AWS API, AWS CLI, atau AWS Management Console untuk melakukan operasi, seperti membuat ember di Amazon S3. Saat Anda melakukannya, Anda mengirimkan sebuah permintaan untuk operasi tersebut. Permintaan Anda menentukan tindakan, sumber daya, sebuah entitas prinsipal (pengguna atau peran), sebuah akun prinsipal, dan informasi permintaan yang diperlukan. Semua informasi ini memberikan konteks.

AWS kemudian memeriksa bahwa Anda (entitas utama) diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk melakukan tindakan yang ditentukan pada sumber daya yang ditentukan. Selama otorisasi, AWS periksa semua kebijakan yang berlaku untuk konteks permintaan Anda. Sebagian besar kebijakan disimpan AWS sebagai JSONdokumen dan menentukan izin untuk entitas utama. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

AWS mengotorisasi permintaan hanya jika setiap bagian dari permintaan Anda diizinkan oleh kebijakan. Untuk melihat diagram dan mempelajari lebih lanjut tentang IAM infrastruktur, lihatCara kerja IAM. Untuk detail tentang cara IAM menentukan apakah permintaan diizinkan, lihatLogika evaluasi kebijakan.

Tag adalah pertimbangan lain dalam proses ini karena tag dapat dilampirkan ke sumber daya atau diteruskan dalam permintaan ke layanan yang mendukung penandaan. Untuk mengontrol akses berdasarkan tanda, Anda memberikan informasi tanda di elemen ketentuan dari kebijakan. Untuk mempelajari apakah suatu AWS layanan mendukung pengendalian akses menggunakan tag, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di ABACkolom. Pilih nama layanan untuk melihat dokumentasi otorisasi dan kontrol akses untuk layanan tersebut.

Anda kemudian dapat membuat IAM kebijakan yang mengizinkan atau menolak akses ke sumber daya berdasarkan tag sumber daya tersebut. Dalam kebijakan tersebut, Anda dapat menggunakan kunci ketentuan tanda untuk mengontrol akses ke salah satu hal berikut:

- Sumber Daya — Kontrol akses ke sumber daya AWS layanan berdasarkan tag pada sumber daya tersebut. Untuk melakukan ini, gunakan aws:ResourceTag/***key-name***Kunci Ketentuan untuk menentukan apakah akan memungkinkan akses ke sumber daya berdasarkan tag yang melekat pada sumber daya.

- Permintaan – Mengontrol tag apa yang dapat diteruskan dalam permintaan. Untuk melakukan ini, gunakan aws:RequestTag/*key-name*kunci kondisi untuk menentukan pasangan nilai kunci tag apa yang dapat diteruskan dalam permintaan untuk menandai sumber daya. AWS
- Setiap bagian dari proses otorisasi — Gunakan aws: TagKeys condition key untuk mengontrol apakah kunci tag tertentu dapat berada dalam permintaan.

Anda dapat membuat IAM kebijakan secara visual, menggunakanJSON, atau dengan mengimpor kebijakan terkelola yang ada. Untuk detailnya, lihat Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan.

> **Note**
>
> Beberapa layanan memungkinkan pengguna untuk menentukan tag ketika mereka membuat sumber daya jika mereka memiliki izin untuk menggunakan tindakan yang membuat sumber daya.

Mengontrol akses ke sumber daya AWS

Anda dapat menggunakan kondisi dalam IAM kebijakan Anda untuk mengontrol akses ke AWS sumber daya berdasarkan tag pada sumber daya tersebut. Anda dapat melakukannya menggunakan kunci syarat `aws:ResourceTag/`*`tag-key`*, atau kunci syarat layanan khusus. Beberapa layanan hanya mendukung versi khusus layanan kunci ini dan bukan versi global.

> **Warning**
>
> Jangan mencoba mengontrol siapa yang dapat melewati peran dengan menandai peran dan kemudian menggunakan kunci `ResourceTag` kondisi dalam kebijakan dengan `iam:PassRole` tindakan tersebut. Pendekatan ini tidak memiliki hasil yang dapat diandalkan. Untuk informasi lebih lanjut tentang izin yang diperlukan untuk memberikan sebuah peran ke sebuah layanan, lihat Berikan izin pengguna untuk meneruskan peran ke layanan AWS.

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan memulai atau menghentikan instans Amazon. EC2 Operasi ini hanya diperbolehkan jika tag instance `Owner` memiliki nilai nama pengguna. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

Anda dapat melampirkan kebijakan ini ke IAM pengguna di akun Anda. Jika pengguna bernama `richard-roe` mencoba memulai EC2 instans Amazon, instance tersebut harus diberi tag `Owner=richard-roe` atau`owner=richard-roe`. Atau, aksesnya akan ditolak. Kunci tanda `Owner` cocok dengan kedua `Owner` dan `owner` karena nama kunci ketentuan tidak terpengaruh huruf besar/kecil. Untuk informasi selengkapnya, lihat IAMJSONelemen kebijakan: Condition.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menggunakan tag `team` utama dalam sumber daya. ARN Kebijakan memberikan izin untuk menghapus antrean Amazon Simple Queue Service, tetapi hanya jika nama antrian dimulai dengan nama tim yang diikuti. `-queue` Misalnya, `qa-queue` if `qa` adalah nama tim untuk tag `team` utama.

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2::${aws:PrincipalTag/team}-queue"
      }
```

```
}
```

Mengontrol akses selama permintaan AWS

Anda dapat menggunakan kondisi dalam IAM kebijakan Anda untuk mengontrol pasangan nilai kunci tag apa yang dapat diteruskan dalam permintaan yang menerapkan tag ke sumber daya. AWS

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan penggunaan EC2 `CreateTags` tindakan Amazon untuk melampirkan tag ke instance. Anda dapat memberikan tanda hanya jika tanda berisi kunci `environment` dan `preprod` atau nilai `production`. Jika anda mau, Anda bisa menggunakan pengubah `ForAllValues` dengan kunci ketentuan `aws:TagKeys` untuk menunjukkan bahwa hanya kunci `environment` yang diperbolehkan dalam permintaan. Ini menghentikan pengguna untuk memasukkan kunci lain, seperti tidak sengaja menggunakan `Environment` daripada `environment`.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

Mengontrol akses berdasarkan kunci tanda

Anda dapat menggunakan kondisi dalam IAM kebijakan Anda untuk mengontrol apakah kunci tag tertentu dapat digunakan dalam permintaan.

Kami menyarankan bahwa ketika Anda menggunakan kebijakan untuk mengontrol akses menggunakan tag, Anda menggunakan tombol `aws:TagKeys` kondisi. AWS layanan yang

mendukung tag memungkinkan Anda membuat beberapa nama kunci tag yang hanya berbeda berdasarkan kasus, seperti menandai EC2 instance Amazon dengan `stack=production` dan`Stack=test`. Nama-nama kunci tidak peka huruf dalam syarat kebijakan. Ini berarti jika Anda menentukan `"aws:ResourceTag/TagKey1": "Value1"` dalam elemen ketentuan kebijakan Anda, kemudian ketentuan tersebut cocok dengan kunci tanda sumber daya bernama `TagKey1` atau `tagkey1`, tetapi tidak keduanya. Untuk mencegah tag duplikat dengan kunci yang hanya bervariasi menurut kasus, gunakan `aws:TagKeys` kondisi untuk menentukan kunci tag yang dapat diterapkan pengguna Anda, atau gunakan kebijakan tag, yang AWS Organizations tersedia. Untuk informasi selengkapnya, lihat Kebijakan Tag di Panduan Pengguna Organizations.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pembuatan dan penandaan rahasia Secrets Manager, tetapi hanya dengan kunci tag atau. `environment` `cost-center` `Null`Kondisi memastikan bahwa kondisi mengevaluasi `false` jika tidak ada tag dalam permintaan.

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "environment",
                    "cost-center"
                ]
            }
        }
}
```

Akses sumber daya lintas akun di IAM

Untuk beberapa AWS layanan, Anda dapat memberikan akses lintas akun ke sumber daya Anda menggunakanIAM. Untuk melakukannya, Anda dapat melampirkan kebijakan sumber daya langsung ke sumber daya yang ingin Anda bagikan, atau menggunakan peran sebagai proxy.

Untuk berbagi sumber daya secara langsung, sumber daya yang ingin Anda bagikan harus mendukung kebijakan berbasis sumber daya. Tidak seperti kebijakan berbasis identitas untuk peran, kebijakan berbasis sumber daya menentukan siapa (prinsipal mana) yang dapat mengakses sumber daya tersebut.

Gunakan peran sebagai proxy saat Anda ingin mengakses sumber daya di akun lain yang tidak mendukung kebijakan berbasis sumber daya.

Untuk detail tentang perbedaan antara jenis kebijakan ini, lihatKebijakan berbasis identitas dan kebijakan berbasis sumber daya.

> **Note**
>
> IAMperan dan kebijakan berbasis sumber daya mendelegasikan akses di seluruh akun hanya dalam satu partisi. Misalnya, Anda memiliki akun di AS Barat (California N.) di `aws` partisi standar. Anda juga memiliki akun di China di `aws-cn` partisi. Anda tidak dapat menggunakan kebijakan berbasis sumber daya di akun Anda di Indonesia untuk mengizinkan akses bagi pengguna di akun standar Anda. AWS

Akses lintas akun menggunakan peran

Tidak semua AWS layanan mendukung kebijakan berbasis sumber daya. Untuk layanan ini, Anda dapat menggunakan IAM peran lintas akun untuk memusatkan manajemen izin saat menyediakan akses lintas akun ke beberapa layanan. Peran lintas akun adalah IAM IAM peran yang mencakup kebijakan kepercayaan yang memungkinkan IAM kepala sekolah di AWS akun lain untuk mengambil peran tersebut. Sederhananya, Anda dapat membuat peran dalam satu AWS akun yang mendelegasikan izin tertentu ke akun lain AWS .

Untuk informasi tentang melampirkan kebijakan ke IAM identitas, lihatKelola IAM kebijakan.

> **Note**
>
> Saat prinsipal beralih ke peran untuk sementara menggunakan izin peran, mereka melepaskan izin aslinya dan mengambil izin yang ditetapkan ke peran yang diasumsikan.

Mari kita lihat keseluruhan proses yang berlaku untuk perangkat lunak APN Mitra yang perlu mengakses akun pelanggan.

1. Pelanggan membuat IAM peran di akun mereka sendiri dengan kebijakan yang memungkinkan akses sumber daya Amazon S3 yang dibutuhkan APN mitra. Dalam contoh ini, nama peran adalah`APNPartner`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

2. Kemudian, pelanggan menentukan bahwa peran tersebut dapat diasumsikan oleh AWS akun mitra dengan memberikan Akun AWS ID APN Mitra dalam kebijakan kepercayaan untuk `APNPartner` peran tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::APN-account-ID:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. Pelanggan memberikan Nama Sumber Daya Amazon (ARN) peran tersebut kepada APN mitra. `ARN`Itu adalah nama peran yang sepenuhnya memenuhi syarat.

```
arn:aws:iam::APN-ACCOUNT-ID:role/APNPartner
```

> **Note**
>
> Sebaiknya gunakan ID eksternal dalam situasi multi-penyewa. Untuk detailnya, lihat Akses ke Akun AWS yang dimiliki oleh pihak ketiga.

4. Ketika perangkat lunak APN Mitra perlu mengakses akun pelanggan, perangkat lunak memanggil bagian AssumeRoleAPIdalam AWS Security Token Service dengan peran dalam akun pelanggan. ARN STSmengembalikan AWS kredensi sementara yang memungkinkan perangkat lunak untuk melakukan tugasnya.

Untuk contoh lain pemberian akses lintas akun menggunakan peran, lihat. Akses untuk IAM pengguna lain Akun AWS yang Anda miliki Anda juga dapat mengikutiIAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran.

Akses lintas akun menggunakan kebijakan berbasis sumber daya

Saat akun mengakses sumber daya melalui akun lain menggunakan kebijakan berbasis sumber daya, prinsipal masih berfungsi di akun tepercaya dan tidak harus melepaskan izinnya untuk menerima izin peran. Dengan kata lain, kepala sekolah terus memiliki akses ke sumber daya di akun tepercaya sambil memiliki akses ke sumber daya di akun kepercayaan. Ini berguna untuk tugas seperti menyalin informasi ke atau dari sumber daya yang dibagikan di akun lain.

Prinsipal yang dapat Anda tentukan dalam kebijakan berbasis sumber daya mencakup akun, IAM pengguna, pengguna gabungan, peran, sesi IAM peran yang diasumsikan, atau layanan. AWS Untuk informasi selengkapnya, lihat Menentukan prinsipal.

Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal.

Daftar berikut mencakup beberapa AWS layanan yang mendukung kebijakan berbasis sumber daya. Untuk daftar lengkap dari semakin banyak AWS layanan yang mendukung melampirkan kebijakan izin ke sumber daya, bukan prinsipal, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Berbasis Sumber Daya.

- Bucket Amazon S3 — Kebijakan ini dilampirkan ke bucket, tetapi kebijakan mengontrol akses ke bucket dan objek di dalamnya. Untuk informasi selengkapnya, lihat Kebijakan Bucket untuk Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon. Dalam beberapa

kasus, yang terbaik mungkin adalah menggunakan peran untuk akses lintas akun ke Amazon S3 Untuk informasi selengkapnya, lihat contoh penelusuran di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

- Topik Amazon Simple Notification Service (AmazonSNS) — Untuk informasi selengkapnya, buka Contoh kasus untuk kontrol SNS akses Amazon di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.
- Antrian Layanan Antrian Sederhana Amazon (AmazonSQS) - Untuk informasi selengkapnya, buka Lampiran: Bahasa Kebijakan Akses di Panduan Pengembang Layanan Antrian Sederhana Amazon.

Kebijakan berbasis sumber daya untuk mendelegasikan izin AWS

Jika sumber daya memberikan izin kepada prinsipal di akun Anda, Anda dapat mendelegasikan izin tersebut ke identitas tertentu. IAM Identitas adalah pengguna, kelompok pengguna, atau peran dalam akun Anda. Anda mendelegasikan izin dengan melampirkan kebijakan ke identitas. Anda dapat memberikan izin maksimum yang diizinkan oleh akun pemilik sumber daya.

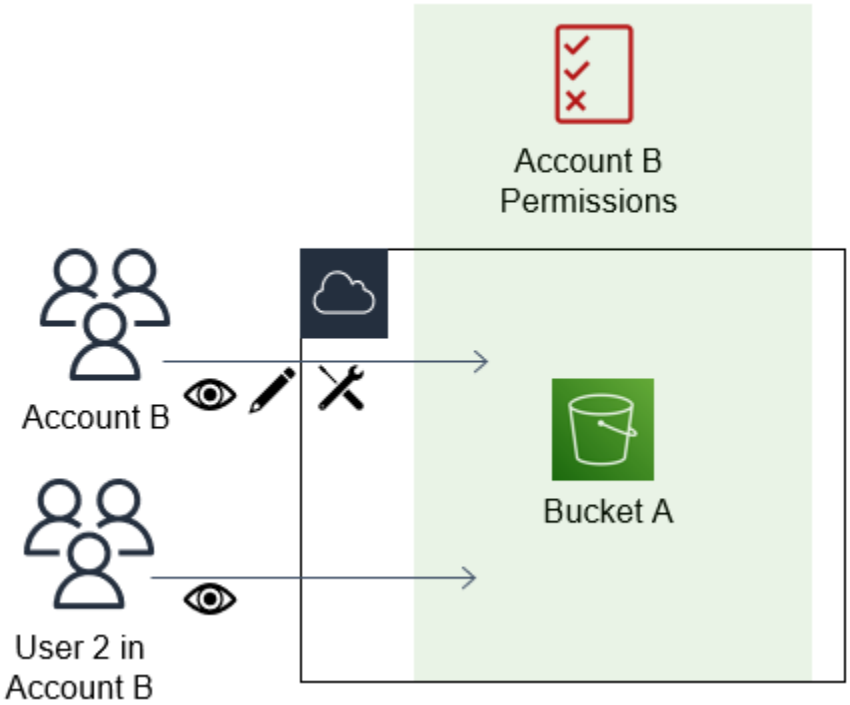
> **Important**
>
> Dalam akses lintas akun, prinsipal membutuhkan `Allow` kebijakan identitas dan kebijakan berbasis sumber daya.

Asumsikan bahwa kebijakan berbasis sumber daya memberi semua prinsipal di akun Anda akses administratif penuh ke sumber daya. Kemudian Anda dapat mendelegasikan akses penuh, akses hanya-baca, atau akses sebagian lainnya ke kepala sekolah di akun Anda. AWS Alternatifnya, jika kebijakan berbasis sumber daya hanya memberikan izin daftar, maka Anda hanya dapat mendelegasikan akses daftar. Jika Anda mencoba untuk mendelegasikan lebih banyak izin dari yang dimiliki akun Anda, prinsipal Anda masih hanya akan memiliki akses daftar.

Untuk informasi selengkapnya tentang bagaimana keputusan ini dibuat, lihat Menentukan apakah permintaan diizinkan atau ditolak dalam akun.

> **Note**
>
> IAMperan dan kebijakan berbasis sumber daya mendelegasikan akses di seluruh akun hanya dalam satu partisi. Misalnya, Anda tidak dapat menambahkan akses lintas akun antara akun dalam partisi `aws` standar dan akun dalam partisi `aws-cn`.

Sebagai contoh, anggaplah bahwa Anda mengelola `AccountA` dan `AccountB`. Di Accounta, Anda memiliki bucket Amazon S3 bernama. `BucketA`



1. Anda melampirkan kebijakan berbasis sumber daya `BucketA` yang memungkinkan semua prinsipal di accountB akses penuh ke objek di bucket Anda. Mereka dapat membuat, membaca, atau menghapus objek apa pun dalam bucket tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::AccountB:root"},
            "Action": "s3:*",
```

```
            "Resource": "arn:aws:s3:::BucketA/*"
            }
        ]
}
```

AccountA memberikan accountB akses penuh ke BucketA dengan menamai accounTB sebagai prinsipal dalam kebijakan berbasis sumber daya. Akibatnya, accounTB berwenang untuk melakukan tindakan apa pun pada BuckETA, dan administrator accountB dapat mendelegasikan akses ke penggunanya di AccounTB.

Pengguna root accountB memiliki semua izin yang diberikan ke akun. Oleh karena itu, pengguna root memiliki akses penuh ke BuckEta.

2. Di accountB, lampirkan kebijakan ke IAM pengguna bernama User2. Kebijakan tersebut memungkinkan pengguna akses hanya-baca ke objek di BuckEta. Itu berarti bahwa User2 dapat melihat objek, tetapi tidak membuat, mengedit, atau menghapusnya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Action" : [
                "s3:Get*",
                "s3:List*" ],
                "Resource" : "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

Tingkat akses maksimum yang dapat didelegasikan accountB adalah tingkat akses yang diberikan ke akun. Dalam hal ini, kebijakan berbasis sumber daya memberikan akses penuh ke accounTB, tetapi User2 hanya diberikan akses read-only.

Administrator AccountB tidak memberikan akses ke User1. Secara default, pengguna tidak memiliki izin apa pun kecuali yang diberikan secara eksplisit, sehingga User1 tidak memiliki akses ke BuckEta.

IAMmengevaluasi izin kepala sekolah pada saat kepala sekolah membuat permintaan. Jika Anda menggunakan wildcard (*) untuk memberi pengguna akses penuh ke sumber daya Anda, prinsipal

dapat mengakses sumber daya apa pun yang dapat diakses AWS akun Anda. Hal ini berlaku bahkan untuk sumber daya yang Anda tambahkan atau Anda memiliki akses kepadanya setelah membuat kebijakan pengguna.

Dalam contoh sebelumnya, jika accounTB telah melampirkan kebijakan ke User2 yang memungkinkan akses penuh ke semua sumber daya di semua akun, User2 akan secara otomatis memiliki akses ke sumber daya apa pun yang dapat diakses oleh AccounTB. Ini termasuk akses BuckEta dan akses ke sumber daya lain yang diberikan oleh kebijakan berbasis sumber daya di Accounta.

Untuk informasi selengkapnya tentang penggunaan peran yang kompleks, seperti memberikan akses ke aplikasi dan layanan, lihatSkenario umum untuk IAM peran.

> **Important**
>
> Berikan akses hanya kepada entitas yang Anda percayai, dan berikan tingkat akses minimum yang diperlukan. Setiap kali entitas tepercaya adalah AWS akun lain, IAM prinsipal apa pun dapat diberikan akses ke sumber daya Anda. AWS Akun tepercaya dapat mendelegasikan akses hanya sejauh telah diberikan akses; itu tidak dapat mendelegasikan lebih banyak akses daripada akun itu sendiri telah diberikan.

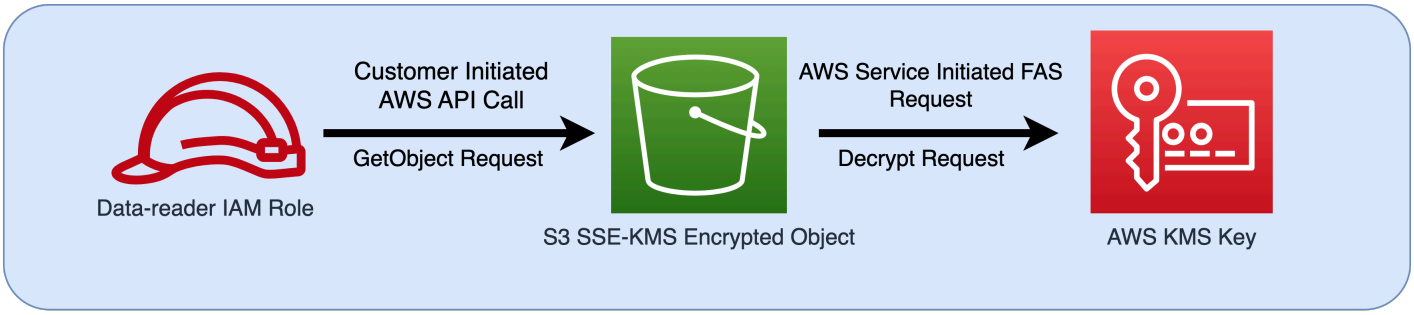
Untuk informasi tentang izin, kebijakan, dan bahasa kebijakan izin yang Anda gunakan untuk menyusun kebijakan, lihat Manajemen akses untuk AWS sumber daya.

Teruskan sesi akses

Forward Access Sessions (FAS) adalah teknologi IAM yang digunakan oleh AWS layanan untuk meneruskan identitas, izin, dan atribut sesi Anda saat AWS layanan membuat permintaan atas nama Anda. FAS menggunakan izin identitas yang memanggil AWS layanan, dikombinasikan dengan identitas AWS layanan untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat untuk AWS layanan atas nama kepala IAM setelah layanan menerima permintaan yang memerlukan interaksi dengan AWS layanan atau sumber daya lain untuk diselesaikan. Ketika permintaan FAS dibuat:

- Layanan yang menerima permintaan awal dari kepala IAM memeriksa izin dari prinsipal IAM.
- Layanan yang menerima permintaan FAS berikutnya juga memeriksa izin dari prinsipal IAM yang sama.

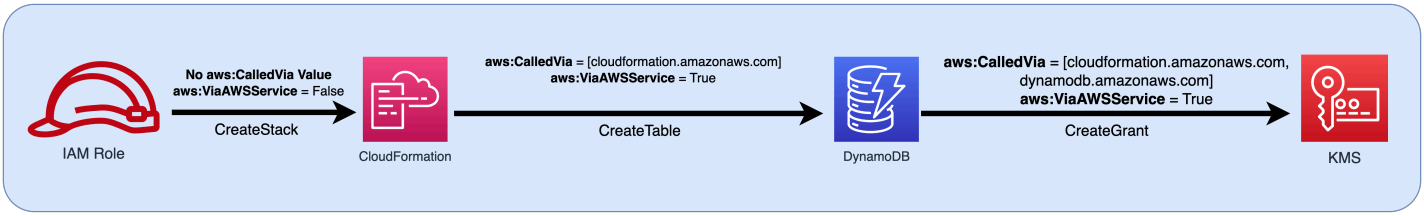
Misalnya, FAS digunakan oleh Amazon S3 untuk melakukan panggilan AWS Key Management Service untuk mendekripsi objek ketika SSE-KMS digunakan untuk mengenkripsinya. Saat mengunduh objek terenkripsi SSE-KMS, peran bernama pembaca data memanggil GetObject objek terhadap Amazon S3, dan tidak memanggil secara langsung. AWS KMS Setelah menerima GetObject permintaan dan mengotorisasi pembaca data, Amazon S3 kemudian membuat permintaan FAS untuk AWS KMS mendekripsi objek Amazon S3. Ketika KMS menerima permintaan FAS, ia memeriksa izin peran dan hanya mengotorisasi permintaan dekripsi jika pembaca data memiliki izin yang benar pada kunci KMS. Permintaan ke Amazon S3 dan AWS KMS diotorisasi menggunakan izin peran dan hanya berhasil jika pembaca data memiliki izin untuk objek Amazon S3 dan kunci KMS. AWS



> **Note**
>
> Permintaan FAS tambahan dapat dilakukan oleh layanan yang telah menerima permintaan FAS. Dalam kasus seperti itu, kepala sekolah yang meminta harus memiliki izin untuk semua layanan yang dipanggil oleh FAS.

Permintaan FAS dan ketentuan kebijakan IAM

Ketika permintaan FAS dibuat,aws: CalledVia,aws: CalledViaFirst, dan kunci aws: CalledViaLast kondisi diisi dengan prinsip layanan layanan yang memulai panggilan FAS. Nilai kunci AWS: V iaAWSService kondisi diatur ke `true` setiap kali permintaan FAS dibuat. Dalam diagram berikut, permintaan untuk CloudFormation secara langsung tidak memiliki `aws:CalledVia` atau kunci `aws:ViaAWSService` kondisi yang ditetapkan. Kapan CloudFormation dan DynamoDB membuat permintaan FAS hilir atas nama peran, nilai untuk kunci kondisi ini diisi.



Untuk mengizinkan permintaan FAS dibuat jika tidak akan ditolak oleh pernyataan kebijakan Deny dengan kunci kondisi yang menguji alamat IP Sumber atau VPC Sumber, Anda harus menggunakan kunci kondisi untuk memberikan pengecualian untuk permintaan FAS dalam kebijakan Deny Anda. Ini dapat dilakukan untuk semua permintaan FAS dengan menggunakan kunci `aws:ViaAWSService` kondisi. Untuk mengizinkan hanya AWS layanan tertentu untuk membuat permintaan FAS, gunakan`aws:CalledVia`.

> **Important**
>
> Ketika permintaan FAS dibuat setelah permintaan awal dibuat melalui titik akhir VPC, nilai kunci kondisi `aws: SourceVpce` untuk,`aws: SourceVpc`, `aws: VpcSourceIp` dan dari permintaan awal tidak digunakan dalam permintaan FAS. Saat menulis kebijakan menggunakan `aws:VPCSourceIP` atau `aws:SourceVPCE` memberikan akses secara kondisional, Anda juga harus menggunakan `aws:ViaAWSService` atau `aws:CalledVia` mengizinkan permintaan FAS. Ketika permintaan FAS dibuat setelah permintaan awal diterima oleh titik akhir AWS layanan publik, permintaan FAS berikutnya akan dibuat dengan nilai kunci `aws:SourceIP` kondisi yang sama.

Contoh: Izinkan akses Amazon S3 dari VPC atau dengan FAS

Dalam contoh kebijakan IAM berikut, permintaan Amazon GetObject S3 dan Athena hanya diizinkan jika permintaan tersebut berasal dari titik akhir VPC yang dilampirkan *ke* example_vpc, atau jika permintaan tersebut adalah permintaan FAS yang dibuat oleh Athena.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
```

```
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
            "example_vpc"
          ]
        }
      }
    },
    {
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

Untuk contoh tambahan menggunakan kunci kondisi untuk mengizinkan akses FAS, lihat contoh repo kebijakan contoh perimeter data.

Contoh kebijakan berbasis identitas IAM

Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika kepala sekolah IAM (pengguna atau peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON yang dilampirkan ke identitas IAM (pengguna, grup pengguna, atau peran). Kebijakan berbasis identitas dapat mencakup kebijakan terkelola AWS , kebijakan terkelola pelanggan, dan kebijakan inline. Untuk mempelajari cara

membuat kebijakan IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan JSON editor”.

Secara default semua permintaan ditolak, jadi Anda harus memberikan akses ke layanan, tindakan, dan sumber daya yang Anda inginkan bagi identitas untuk mengakses. Jika Anda juga ingin mengizinkan akses untuk menyelesaikan tindakan yang ditentukan di konsol IAM, Anda perlu memberikan izin tambahan.

Pustaka kebijakan berikut dapat membantu Anda menentukan izin untuk identitas IAM Anda. Setelah anda menemukan kebijakan yang Anda perlukan, pilih lihat kebijakan ini untuk melihat JSON untuk kebijakan tersebut. Anda dapat menggunakan dokumen kebijakan JSON sebagai contoh untuk kebijakan Anda sendiri.

> **Note**
>
> Jika Anda ingin memberikan sebuah kebijakan untuk disertakan dalam panduan referensi ini, gunakan tombol Umpan Balik di bagian bawah halaman ini.

Contoh kebijakan: AWS

- Mengizinkan akses selama rentang tanggal tertentu. (Lihat kebijakan ini.)
- Memungkinkan mengaktifkan dan menonaktifkan Wilayah AWS . (Lihat kebijakan ini.)
- Memungkinkan pengguna yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan. (Lihat kebijakan ini.)
- Memungkinkan akses khusus saat menggunakan MFA selama rentang tanggal tertentu. (Lihat kebijakan ini.)
- Memungkinkan pengguna mengelola kredensialnya sendiri di halaman Kredensial Keamanan. (Lihat kebijakan ini.)
- Memungkinkan pengguna untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan. (Lihat kebijakan ini.)
- Memungkinkan pengguna untuk mengelola kata sandi mereka sendiri di halaman Kredensial keamanan. (Lihat kebijakan ini.)
- Memungkinkan pengguna mengelola kata sandi, kunci akses, dan kunci publik SSH mereka sendiri di halaman kredensi Keamanan. (Lihat kebijakan ini.)
- Menolak akses AWS berdasarkan Wilayah yang diminta. (Lihat kebijakan ini.)

- Menolak akses AWS berdasarkan alamat IP sumber. (Lihat kebijakan ini.)

Contoh kebijakan: AWS Data Exchange

- Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali. AWS Data Exchange(Lihat kebijakan ini.)

Contoh kebijakan: AWS Data Pipeline

- Menolak akses ke saluran pipa yang tidak dibuat pengguna (Lihat kebijakan ini.)

Contoh kebijakan: Amazon DynamoDB

- Mengizinkan akses ke tabel Amazon DynamoDB tertentu (Lihat kebijakan ini.)
- Mengizinkan akses ke atribut Amazon DynamoDB tertentu (Lihat kebijakan ini.)
- Mengizinkan akses tingkat item ke Amazon DynamoDB berdasarkan ID Amazon Cognito (Lihat kebijakan ini.)

Contoh kebijakan: Amazon EC2

- Mengizinkan melampirkan atau melepaskan volume Amazon EBS ke instans Amazon EC2 berdasarkan tag (Lihat kebijakan ini.)
- Mengizinkan peluncuran instans Amazon EC2 dalam subnet tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pengelolaan grup keamanan Amazon EC2 terkait dengan VPC tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan memulai atau menghentikan instans Amazon EC2 yang telah ditandai pengguna, secara terprogram, dan di konsol (Lihat kebijakan ini.)
- Mengizinkan memulai atau menghentikan instans Amazon EC2 berbasis sumber daya dan tag prinsipal, secara terprogram, dan di konsol (Lihat kebijakan ini.)
- Mengizinkan memulai atau menghentikan instans Amazon EC2 ketika sumber daya dan tag prinsipal cocok (Lihat kebijakan ini.)
- Mengizinkan akses penuh Amazon EC2 dalam Wilayah tertentu, secara terprogram dan di konsol. (Lihat kebijakan ini.)

- Mengizinkan memulai atau menghentikan instans Amazon EC2 tertentu dan mengubah grup keamanan tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Menolak akses ke operasi Amazon EC2 tertentu tanpa MFA (Lihat kebijakan ini.)
- Membatasi akhir instans Amazon EC2 ke rentang alamat IP tertentu (Lihat kebijakan ini.)

Contoh kebijakan: AWS Identity and Access Management (IAM)

- Mengizinkan akses ke API simulator kebijakan (Lihat kebijakan ini.)
- Mengizinkan akses ke konsol simulator kebijakan (Lihat kebijakan ini.)
- Mengizinkan untuk memberikan peran apa pun yang memiliki tanda tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan dan menolak akses ke beberapa layanan, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan penambahan tag spesifik ke pengguna IAM dengan sebuah tag spesifik yang berbeda, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan penambahan tag spesifik ke setiap pengguna IAM atau peran, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pembuatan pengguna baru hanya dengan tag spesifik (Lihat kebijakan ini.)
- Mengizinkan menghasilkan dan mengambil laporan kredensial IAM (Lihat kebijakan ini.)
- Mengizinkan pengelolaan keanggotaan grup, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pengelolaan tag tertentu (Lihat kebijakan ini.)
- Mengizinkan untuk memberikan sebuah peran IAM ke layanan tertentu (Lihat kebijakan ini.)
- Mengizinkan akses hanya-baca ke konsol IAM tanpa melaporkan (Lihat kebijakan ini.)
- Mengizinkan akses hanya-baca ke konsol IAM (Lihat kebijakan ini.)
- Mengizinkan pengguna spesifik untuk mengelola sebuah grup, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pengaturan persyaratan kata sandi akun, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan penggunaan API simulator kebijakan untuk pengguna dengan jalan tertentu (Lihat kebijakan ini.)
- Mengizinkan penggunaan konsol simulator kebijakan untuk pengguna dengan jalan tertentu (Lihat kebijakan ini.)
- Mengizinkan pengguna IAM untuk mengatur sendiri perangkat MFA. (Lihat kebijakan ini.)

- Memungkinkan pengguna IAM untuk mengatur kredensialnya sendiri, secara terprogram dan di konsol. (Lihat kebijakan ini.)
- Memungkinkan layanan tampilan informasi yang terakhir diakses untuk AWS Organizations kebijakan di konsol IAM. (Lihat kebijakan ini.)
- Batasi kebijakan terkelola yang dapat diterapkan ke pengguna, kelompok atau peran IAM (Lihat kebijakan ini.)
- Mengizinkan akses ke kebijakan IAM hanya di akun Anda (Lihat kebijakan ini.)

Contoh kebijakan: AWS Lambda

- Mengizinkan AWS Lambda fungsi mengakses tabel Amazon DynamoDB (Lihat kebijakan ini.)

Contoh kebijakan: Amazon RDS

- Mengizinkan akses basis data RDS sepenuhnya di dalam Wilayah tertentu (Lihat kebijakan ini.)
- Mengizinkan pemulihan basis data Amazon RDS, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pemilik tag mengakses sepenuhnya sumber daya Amazon RDS yang telah mereka tandai (Lihat kebijakan ini.)

Contoh kebijakan: Amazon S3

- Mengizinkan pengguna Amazon Cognito mengakses objek dalam bucket Amazon S3 milik mereka (Lihat kebijakan ini.)
- Mengizinkan pengguna gabungan untuk mengakses direktori home milik mereka di Amazon S3, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan akses S3 penuh, tetapi secara tegas menolak akses ke bucket Produksi jika administrator belum masuk menggunakan MFA dalam tiga puluh menit terakhir (Lihat kebijakan ini.)
- Mengizinkan pengguna IAM untuk mengakses direktori home milik mereka di Amazon S3, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Memungkinkan pengguna mengelola satu bucket Amazon S3 dan menolak setiap AWS tindakan dan sumber daya lainnya (Lihat kebijakan ini.)
- Mengizinkan akses `Read` dan `Write` ke bucket Amazon S3 tertentu (Lihat kebijakan ini.)

- Mengizinkan akses `Read` dan `Write` ke bucket Amazon S3 tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)

AWS: Mengizinkan akses berdasarkan tanggal dan waktu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan akses ke tindakan berdasarkan tanggal dan waktu. Kebijakan ini membatasi akses ke tindakan yang terjadi antara 1 April 2020 dan 30 Juni 2020 (UTC), inklusif. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk mempelajari tentang menggunakan beberapa ketentuan dalam blok `Condition` kebijakan IAM, lihat Beberapa nilai dalam suatu ketentuan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
            }
        }
    ]
}
```

> **Note**
>
> Anda tidak dapat menggunakan variabel kebijakan dengan operator kondisi Tanggal. Untuk mempelajari lebih lanjut lihat Elemen kondisi

AWS: Memungkinkan mengaktifkan dan menonaktifkan Wilayah AWS

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan administrator mengaktifkan dan menonaktifkan Wilayah Asia Pasifik (Hong Kong)

(ap-east-1). Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Pengaturan ini muncul di halaman Pengaturan akun di AWS Management Console. Halaman ini mencakup informasi tingkat akun yang sensitif yang harus dilihat dan dikelola hanya oleh administrator akun. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

> **Important**
>
> Anda tidak dapat mengaktifkan atau menonaktifkan wilayah yang diaktifkan secara default. Anda hanya dapat memasukkan wilayah yang dinonaktifkan secara default. Untuk informasi selengkapnya, lihat Mengelola Wilayah AWS di Referensi Umum AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM yang diautentikasi menggunakan otentikasi multi-faktor (MFA) untuk mengelola kredensialnya sendiri di halaman Kredensi Keamanan. Halaman AWS Management Console ini menampilkan informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengubah kata sandi, access key, perangkat MFA, sertifikat X.509, dan kunci SSH serta kredensial Git milik mereka. Contoh kebijakan ini mencakup izin yang diperlukan untuk melihat dan mengubah semua informasi di halaman tersebut. Ini juga mengharuskan pengguna untuk mengatur dan mengautentikasi menggunakan MFA sebelum melakukan operasi lain di. AWS Untuk mengizinkan pengguna mengelola kredensial milik mereka tanpa menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan.

Untuk mempelajari cara pengguna dapat mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

> **Note**
>
> - Kebijakan contoh ini tidak mengizinkan pengguna untuk mengatur ulang kata sandi saat masuk AWS Management Console untuk pertama kalinya. Kami menyarankan Anda untuk tidak memberikan izin kepada pengguna baru sampai setelah mereka masuk. Untuk informasi selengkapnya, lihat Bagaimana cara membuat IAM pengguna dengan aman?. Ini juga mencegah pengguna dengan kata sandi yang kedaluwarsa mengatur ulang kata sandi mereka saat masuk. Anda dapat mengizinkan ini dengan menambahkan `iam:ChangePassword` dan `iam:GetAccountPasswordPolicy` ke pernyataan `DenyAllExceptListedIfNoMFA`. Namun, kami tidak merekomendasikan ini karena mengizinkan pengguna untuk mengubah kata sandi mereka tanpa MFA dapat menjadi risiko keamanan.
> - Jika Anda bermaksud menggunakan kebijakan ini untuk akses terprogram, Anda harus menelepon `GetSessionToken`untuk mengautentikasi dengan MFA. Untuk informasi selengkapnya, lihat APIAkses aman dengan MFA.

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat informasi tingkat akun. Izin ini harus ada di dalam pernyataan milik mereka karena hal itu tidak mendukung atau tidak perlu untuk menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`. Pernyataan ini mencakup tindakan berikut yang mengizinkan pengguna melihat informasi spesifik:
  - `GetAccountPasswordPolicy` – Melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
  - `ListVirtualMFADevices` – Melihat detail tentang perangkat MFA virtual yang diaktifkan untuk pengguna.
- Pernyataan `AllowManageOwnPasswords` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.
- Pernyataan `AllowManageOwnAccessKeys` mengizinkan pengguna untuk membuat, memperbarui, dan menghapus access key milik mereka. Pengguna juga dapat mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan.
- Pernyataan `AllowManageOwnSigningCertificates` mengizinkan pengguna untuk mengunggah, memperbarui, dan menghapus sertifikat tanda tangan milik mereka.
- `AllowManageOwnSSHPublicKeys`Pernyataan ini memungkinkan pengguna untuk mengunggah, memperbarui, dan menghapus kunci publik SSH mereka sendiri. CodeCommit
- `AllowManageOwnGitCredentials`Pernyataan ini memungkinkan pengguna untuk membuat, memperbarui, dan menghapus kredensi Git mereka sendiri untuk. CodeCommit
- `AllowManageOwnVirtualMFADevice`Pernyataan tersebut memungkinkan pengguna untuk membuat perangkat MFA virtual mereka sendiri. ARN sumber daya dalam pernyataan ini memungkinkan pengguna untuk membuat perangkat MFA dengan nama apa pun, tetapi pernyataan lain dalam kebijakan hanya mengizinkan pengguna untuk melampirkan perangkat ke pengguna yang saat ini masuk.
- Pernyataan `AllowManageOwnUserMFA` mengizinkan pengguna melihat atau mengelola perangkat MFA virtual, U2F, atau perangkat keras untuk pengguna milik mereka. ARN sumber daya dalam pernyataan ini mengizinkan akses hanya ke pengguna IAM milik pengguna. Pengguna tidak dapat melihat atau mengelola perangkat MFA untuk pengguna lain.
- `DenyAllExceptListedIfNoMFA`Pernyataan tersebut menolak akses ke setiap tindakan di semua AWS layanan, kecuali beberapa tindakan yang terdaftar, tetapi hanya jika pengguna tidak masuk dengan MFA. Pernyataan ini menggunakan kombinasi `"Deny"` dan `"NotAction"` untuk secara tegas menolak akses ke setiap tindakan yang tidak tercantum. Item yang tercantum tidak

ditolak atau diizinkan oleh pernyataan ini. Namun, tindakan tersebut diizinkan oleh pernyataan lain dalam kebijakan tersebut. Untuk informasi lebih lanjut tentang logika pernyataan ini, lihat NotAction dengan Deny. Jika pengguna masuk di dengan MFA, maka pengujian `Condition` gagal dan pernyataan ini tidak menolak tindakan apa pun. Dalam kasus ini, kebijakan atau pernyataan lain untuk pengguna menentukan izin pengguna.

Pernyataan ini memastikan bahwa ketika pengguna belum masuk dengan MFA yang dapat mereka lakukan hanya tindakan yang tercantum. Sebagai tambahan, mereka dapat melakukan tindakan yang tercantum hanya jika pernyataan atau kebijakan lain mengizinkan akses ke tindakan tersebut. Ini tidak mengizinkan pengguna untuk membuat kata sandi saat masuk, karena tindakan `iam:ChangePassword` seharusnya tidak boleh diizinkan tanpa otorisasi MFA.

Versi `...IfExists` dari operator `Bool` memastikan bahwa jika kunci `aws:MultiFactorAuthPresent` hilang, ketentuan menghasilkan sah. Ini berarti bahwa pengguna yang mengakses kredensial jangka panjang sebuah API, seperti kunci akses, tidak diberi akses ke operasi API non-IAM.

Kebijakan ini tidak mengizinkan pengguna untuk melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengizinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo` dan pernyataan `DenyAllExceptListedIfNoMFA`. Ini juga tidak mengizinkan pengguna untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk memungkinkan ini, tambahkan `iam:GetLoginProfile` dan `iam:UpdateLoginProfile` tindakan ke `AllowManageOwnPasswords` pernyataan. Untuk juga mengizinkan pengguna mengubah kata sandi mereka dari halaman pengguna milik mereka tanpa masuk menggunakan MFA, tambahkan tindakan `iam:UpdateLoginProfile` ke pernyataan `DenyAllExceptListedIfNoMFA`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
```

```
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
```

```
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

AWS: Mengizinkan akses tertentu dalam tanggal tertentu menggunakan MFA

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menggunakan beberapa kondisi, yang dievaluasi menggunakan logika. `AND` Ini memungkinkan akses penuh ke layanan bernama `SERVICE-NAME-1`, dan akses ke tindakan `ACTION-NAME-A` dan `ACTION-NAME-B` dalam layanan bernama `SERVICE-NAME-2`. Tindakan ini hanya diizinkan ketika pengguna diautentikasi menggunakan autentikasi multifaktor (MFA). Akses dibatasi untuk tindakan yang terjadi antara 1 Juli 2017 dan 31 Desember 2017 (UTC), termasuk. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk mempelajari tentang menggunakan beberapa ketentuan dalam blok `Condition` kebijakan IAM, lihat Beberapa nilai dalam suatu ketentuan.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
```

```
    }
}
```

AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengelola semua kredensialnya sendiri di halaman kredensial Keamanan. AWS Management Console Halaman ini menampilkan informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengubah kata sandi, access key, sertifikat X.509, kunci SSH, serta kredensial Git milik mereka. Contoh kebijakan ini mencakup izin yang diperlukan untuk melihat dan mengubah semua informasi di halaman tersebut kecuali perangkat MFA pengguna. Untuk mengizinkan pengguna mengelola kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.

Untuk mempelajari cara pengguna dapat mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat informasi tingkat akun. Izin ini harus ada di dalam pernyataan milik mereka karena hal itu tidak mendukung atau tidak perlu untuk menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`. Pernyataan ini mencakup tindakan berikut yang mengizinkan pengguna melihat informasi spesifik:
  - `GetAccountPasswordPolicy` – Melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
  - `GetAccountSummary` – Melihat ID akun dan akun ID pengguna resmi.
- Pernyataan `AllowManageOwnPasswords` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.
- Pernyataan `AllowManageOwnAccessKeys` mengizinkan pengguna untuk membuat, memperbarui, dan menghapus access key milik mereka. Pengguna juga dapat mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan.
- Pernyataan `AllowManageOwnSigningCertificates` mengizinkan pengguna untuk mengunggah, memperbarui, dan menghapus sertifikat tanda tangan milik mereka.

- `AllowManageOwnSSHPublicKeys`Pernyataan ini memungkinkan pengguna untuk mengunggah, memperbarui, dan menghapus kunci publik SSH mereka sendiri. CodeCommit
- `AllowManageOwnGitCredentials`Pernyataan ini memungkinkan pengguna untuk membuat, memperbarui, dan menghapus kredensi Git mereka sendiri untuk. CodeCommit

Kebijakan ini tidak mengizinkan pengguna melihat atau mengelola perangkat MFA milik mereka. Mereka juga tidak dapat melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengijinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo`. Ini juga tidak mengizinkan pengguna untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk mengizinkan ini, tambahkan tindakan `iam:CreateLoginProfile`, `iam:DeleteLoginProfile`, `iam:GetLoginProfile`, and `iam:UpdateLoginProfile` ke pernyataan `AllowManageOwnPasswords`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
```

```
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
```

```
}
```

AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM yang diautentikasi melalui otentikasi multi-faktor (MFA) untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan. AWS Management Console Halaman ini menampilkan informasi akun dan pengguna, tetapi pengguna hanya dapat melihat dan mengedit perangkat MFA mereka sendiri. Untuk mengizinkan pengguna mengelola semua kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.

> **Note**
>
> Jika pengguna IAM dengan kebijakan ini tidak diautentikasi oleh MFA, kebijakan ini menolak akses ke semua AWS tindakan kecuali yang diperlukan untuk mengautentikasi menggunakan MFA. Untuk menggunakan AWS API AWS CLI dan, pengguna IAM harus terlebih dahulu mengambil token MFA mereka menggunakan AWS STS GetSessionTokenoperasi dan kemudian menggunakan token itu untuk mengautentikasi operasi yang diinginkan. Kebijakan lain, seperti kebijakan berbasis sumber daya atau kebijakan berbasis identitas lainnya dapat memungkinkan tindakan di layanan lain. Kebijakan ini akan menolak akses tersebut jika pengguna IAM tidak diautentikasi oleh MFA.

Untuk mempelajari cara pengguna dapat mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat detail tentang perangkat MFA virtual yang diaktifkan untuk pengguna. Izin ini harus berada dalam pernyataannya sendiri karena hal itu tidak mendukung ARN sumber daya tertentu. Sebagai gantinya Anda harus menentukan `"Resource" : "*"`.
- `AllowManageOwnVirtualMFADevice`Pernyataan tersebut memungkinkan pengguna untuk membuat perangkat MFA virtual mereka sendiri. ARN sumber daya dalam pernyataan ini memungkinkan pengguna untuk membuat perangkat MFA dengan nama apa pun, tetapi

pernyataan lain dalam kebijakan hanya mengizinkan pengguna untuk melampirkan perangkat ke pengguna yang saat ini masuk.

- Pernyataan `AllowManageOwnUserMFA` mengizinkan pengguna untuk melihat atau mengelola perangkat MFA virtual, U2F, atau perangkat keras milik mereka. ARN sumber daya dalam pernyataan ini mengizinkan akses hanya ke pengguna IAM milik pengguna. Pengguna tidak dapat melihat atau mengelola perangkat MFA untuk pengguna lain.
- `DenyAllExceptListedIfNoMFA`Pernyataan tersebut menolak akses ke setiap tindakan di semua AWS layanan, kecuali beberapa tindakan yang terdaftar, tetapi hanya jika pengguna tidak masuk dengan MFA. Pernyataan ini menggunakan kombinasi `"Deny"` dan `"NotAction"` untuk secara tegas menolak akses ke setiap tindakan yang tidak tercantum. Item yang tercantum tidak ditolak atau diizinkan oleh pernyataan ini. Namun, tindakan tersebut diizinkan oleh pernyataan lain dalam kebijakan tersebut. Untuk informasi lebih lanjut tentang logika pernyataan ini, lihat NotAction dengan Deny. Jika pengguna masuk di dengan MFA, maka pengujian `Condition` gagal dan pernyataan ini tidak menolak tindakan apa pun. Dalam kasus ini, kebijakan atau pernyataan lain untuk pengguna menentukan izin pengguna.

  Pernyataan ini memastikan bahwa ketika pengguna belum masuk di dengan MFA, mereka hanya dapat melakukan tindakan yang tercantum. Sebagai tambahan, mereka dapat melakukan tindakan yang tercantum hanya jika pernyataan atau kebijakan lain mengizinkan akses ke tindakan tersebut.

  Versi `...IfExists` dari operator `Bool` memastikan bahwa jika kunci `aws:MultiFactorAuthPresent` hilang, ketentuan menghasilkan sah. Ini berarti bahwa pengguna yang mengakses kredensial jangka panjang operasi API, seperti kunci akses, ditolak aksesnya ke operasi API non-IAM.

Kebijakan ini tidak mengizinkan pengguna untuk melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengizinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo` dan pernyataan `DenyAllExceptListedIfNoMFA`.

> **Warning**
>
> Jangan tambahkan izin untuk menghapus perangkat MFA tanpa autentikasi MFA. Pengguna dengan kebijakan ini mungkin mencoba untuk menetapkan sendiri perangkat MFA virtual dan menerima kesalahan bahwa mereka tidak berwenang untuk melakukan. `iam:DeleteVirtualMFADevice` Jika ini terjadi, jangan tambahkan izin tersebut ke pernyataan `DenyAllExceptListedIfNoMFA`. Pengguna yang tidak diautentikasi

menggunakan MFA seharusnya tidak pernah diizinkan untuk menghapus perangkat MFA mereka. Pengguna mungkin melihat kesalahan ini jika sebelumnya mereka mulai memberikan perangkat MFA virtual ke penggunanya dan membatalkan proses tersebut. Untuk mengatasi masalah ini, Anda atau administrator lain harus menghapus perangkat MFA virtual pengguna yang ada menggunakan API AWS CLI atau AWS . Untuk informasi selengkapnya, lihat Saya tidak berwenang untuk melakukan: iam: DeleteVirtual MFADevice.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
```

```
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS: Memungkinkan pengguna IAM untuk mengubah kata sandi konsol mereka sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengubah AWS Management Console kata sandi mereka sendiri di halaman kredensial Keamanan. AWS Management Console Halaman ini menampilkan informasi akun dan pengguna, tetapi pengguna hanya dapat mengakses kata sandi mereka sendiri. Untuk mengizinkan pengguna mengelola semua kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan. Untuk mengizinkan pengguna mengelola kredensial milik mereka tanpa menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan.

Untuk mempelajari cara pengguna dapat mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `ViewAccountPasswordRequirements` mengizinkan pengguna untuk melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
- Pernyataan `ChangeOwnPassword` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.

Kebijakan ini tidak mengizinkan pengguna untuk melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengijinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `ViewAccountPasswordRequirements`. Ini juga tidak mengizinkan pengguna untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk memungkinkan ini, tambahkan `iam:GetLoginProfile` dan `iam:UpdateLoginProfile` tindakan ke `ChangeOwnPasswords` pernyataan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: Memungkinkan pengguna IAM untuk mengelola kata sandi, kunci akses, dan kunci publik SSH mereka sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengelola kata sandi, kunci akses, dan sertifikat X.509 mereka sendiri di halaman kredensial Keamanan. Halaman AWS Management Console ini menampilkan informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengubah kata sandi, access key, perangkat MFA, sertifikat X.509, kunci SSH, dan kredensial Git milik mereka. Kebijakan contoh ini mencakup izin yang diperlukan untuk hanya melihat dan mengubah kata sandi, access key, dan sertifikat X.509 miliknya. Untuk mengizinkan pengguna mengelola semua kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk

mengelola kredensialnya sendiri di halaman kredensi Keamanan. Untuk mengizinkan pengguna mengelola kredensial milik mereka tanpa menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan.

Untuk mempelajari cara pengguna mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat informasi tingkat akun. Izin ini harus ada di dalam pernyataan milik mereka karena hal itu tidak mendukung atau tidak perlu untuk menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`. Pernyataan ini mencakup tindakan berikut yang mengizinkan pengguna melihat informasi spesifik:
  - `GetAccountPasswordPolicy` – Melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
  - `GetAccountSummary` – Melihat ID akun dan akun ID pengguna resmi.
- Pernyataan `AllowManageOwnPasswords` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.
- Pernyataan `AllowManageOwnAccessKeys` mengizinkan pengguna untuk membuat, memperbarui, dan menghapus access key milik mereka. Pengguna juga dapat mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan.
- `AllowManageOwnSSHPublicKeys`Pernyataan ini memungkinkan pengguna untuk mengunggah, memperbarui, dan menghapus kunci publik SSH mereka sendiri. CodeCommit

Kebijakan ini tidak mengizinkan pengguna melihat atau mengelola perangkat MFA milik mereka. Mereka juga tidak dapat melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengijinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo`. Ini juga tidak mengizinkan pengguna untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk memungkinkan ini, tambahkan `iam:GetLoginProfile` dan `iam:UpdateLoginProfile` tindakan ke `AllowManageOwnPasswords` pernyataan.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
```

```
}
```

AWS: Menolak akses AWS berdasarkan Wilayah yang diminta

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang menolak akses ke tindakan apa pun di luar Wilayah yang ditentukan menggunakan kunci `aws:RequestedRegion` kondisi, kecuali untuk tindakan dalam layanan yang ditentukan menggunakan. `NotAction` Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kebijakan ini menggunakan elemen `NotAction` dengan efek `Deny`, yang secara tegas menghalangi akses ke semua tindakan yang tidak tercantum dalam pernyataan. Tindakan dalam CloudFront, IAM, Rute 53, dan AWS Support layanan tidak boleh ditolak karena ini adalah layanan AWS global populer dengan titik akhir tunggal yang secara fisik terletak di `us-east-1` Wilayah. Karena semua permintaan untuk layanan ini diajukan ke Wilayah `us-east-1`, permintaan tidak akan ditolak tanpa elemen `NotAction`. Edit elemen ini untuk menyertakan tindakan untuk layanan AWS global lainnya yang Anda gunakan, sepertibudgets,`globalaccelerator`,`importexport`,`organizations`, atau`waf`. Beberapa layanan global lainnya, seperti AWS Chatbot dan AWS Device Farm, adalah layanan global dengan titik akhir yang secara fisik berlokasi di `us-west-2` wilayah tersebut. Untuk mempelajari tentang semua layanan yang memiliki titik akhir global tunggal, lihatWilayah dan Titik Akhir AWS di Referensi Umum AWS. Untuk informasi selengkapnya tentang penggunaan elemen `NotAction` dengan efek `Deny`, lihat IAMJSONelemen kebijakan: NotAction.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
```

```
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

AWS: Menolak akses AWS berdasarkan IP sumber

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menolak akses ke semua AWS tindakan di akun saat permintaan berasal dari prinsipal di luar rentang IP yang ditentukan. Kebijakan ini berguna saat alamat IP untuk perusahaan Anda berada dalam cakupan tertentu. Dalam contoh ini, permintaan akan ditolak kecuali berasal dari kisaran CIDR 192.0.2.0/24 atau 203.0.113.0/24. Kebijakan ini tidak menolak permintaan yang dibuat oleh AWS layanan yang digunakan Teruskan sesi akses karena alamat IP pemohon asli dipertahankan.

Hati-hati saat menggunakan kondisi negatif dalam pernyataan kebijakan yang sama seperti `"Effect": "Deny"`. Saat Anda melakukannya, tindakan yang ditentukan dalam pernyataan kebijakan secara eksplisit ditolak di semua kondisi kecuali untuk yang ditentukan.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

Ketika kebijakan lain mengizinkan tindakan, prinsipal dapat membuat permintaan dari rentang alamat IP. AWS Layanan juga dapat membuat permintaan menggunakan kredensi kepala sekolah. Saat prinsipal mengajukan permintaan dari luar rentang IP, permintaan tersebut ditolak.

Untuk informasi selengkapnya tentang penggunaan kunci `aws:SourceIp` kondisi, termasuk informasi tentang kapan `aws:SourceIp` mungkin tidak berfungsi dalam kebijakan Anda, lihatAWS kunci konteks kondisi global.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang menolak akses ke semua sumber daya AWS yang bukan milik akun Anda, kecuali sumber daya yang AWS Data Exchange diperlukan untuk pengoperasian normal. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Anda dapat membuat kebijakan serupa untuk membatasi akses ke sumber daya dalam organisasi atau unit organisasi, sambil menghitung sumber daya yang AWS Data Exchange dimiliki dengan menggunakan kunci kondisi `aws:ResourceOrgPaths` dan`aws:ResourceOrgID`.

Jika Anda menggunakan AWS Data Exchange di lingkungan Anda, layanan akan membuat dan berinteraksi dengan sumber daya seperti bucket Amazon S3 yang dimiliki oleh akun layanan. Misalnya, AWS Data Exchange mengirimkan permintaan ke bucket Amazon S3 yang dimiliki oleh AWS Data Exchange layanan atas nama prinsipal IAM (pengguna atau peran) yang menjalankan API. AWS Data Exchange Dalam hal ini, menggunakan`aws:ResourceAccount`,`aws:ResourceOrgPaths`, atau `aws:ResourceOrgID`

dalam kebijakan, tanpa memperhitungkan sumber daya yang AWS Data Exchange dimiliki, menolak akses ke ember yang dimiliki oleh akun layanan.

- Pernyataan tersebut`DenyAllAwsResourcesOutsideAccountExceptS3`,, menggunakan `NotAction` elemen dengan efek Deny yang secara eksplisit menolak akses ke setiap tindakan yang tidak tercantum dalam pernyataan yang juga bukan milik akun yang terdaftar. `NotAction`Elemen menunjukkan pengecualian untuk pernyataan ini. Tindakan ini merupakan pengecualian untuk pernyataan ini karena jika tindakan dilakukan pada sumber daya yang dibuat oleh AWS Data Exchange, kebijakan menyangkalnya.
- Pernyataan itu`DenyAllS3ResoucesOutsideAccountExceptDataExchange`,, menggunakan kombinasi `ResourceAccount` dan `CalledVia` kondisi untuk menolak akses ke tiga tindakan Amazon S3 yang dikecualikan dalam pernyataan sebelumnya. Pernyataan tersebut menyangkal tindakan jika sumber daya tidak termasuk dalam akun yang terdaftar dan jika layanan panggilan tidak AWS Data Exchange. Pernyataan tersebut tidak menyangkal tindakan jika sumber daya milik akun yang terdaftar atau kepala layanan yang terdaftar`dataexchange.amazonaws.com`,, melakukan operasi.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Ini menggunakan `Deny` efek yang secara eksplisit menolak akses ke semua sumber daya yang tercantum dalam pernyataan yang bukan milik akun yang terdaftar. Gunakan kebijakan ini dalam kombinasi dengan kebijakan lain yang memungkinkan akses ke sumber daya tertentu.

Contoh berikut menunjukkan cara mengonfigurasi kebijakan untuk mengizinkan akses ke bucket Amazon S3 yang diperlukan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
```

```
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWS Data Pipeline: Menolak akses ke DataPipeline saluran pipa yang tidak dibuat pengguna

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menolak akses ke pipeline yang tidak dibuat pengguna. Jika nilai kolom `PipelineCreator` cocok dengan nama pengguna IAM, maka tindakan yang ditentukan tidak ditolak. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator": "${aws:userid}"}
            }
        }
    ]
}
```

Amazon DynamoDB: Memungkinkan akses ke tabel tertentu

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan akses penuh ke tabel `MyTable` DynamoDB. Kebijakan ini memberikan izin yang diperlukan untuk

menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

> **Important**
>
> Kebijakan ini memungkinkan semua tindakan yang dapat dilakukan pada tabel DynamoDB. Untuk meninjau tindakan ini, lihat Izin API DynamoDB: Referensi Tindakan, Sumber Daya, dan Ketentuan di Panduan Pengembang Amazon DynamoDB. Anda dapat memberikan izin yang sama dengan mencantumkan setiap tindakan individu. Namun, jika Anda menggunakan wildcard (*) dalam `Action` elemen, misalnya, Anda tidak perlu memperbarui kebijakan jika DynamoDB menambahkan tindakan List baru. `"dynamodb:List*"`

Kebijakan ini mengizinkan tindakan hanya pada tabel DynamoDB yang ada dengan nama yang ditentukan. Untuk memungkinkan pengguna `Read` mengakses semua yang ada di DynamoDB, Anda juga dapat melampirkan AmazonDynamo kebijakan terkelola DB. ReadOnlyAccess AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
                "dynamodb:DescribeTable",
                "dynamodb:Get*",
                "dynamodb:Query",
```

```
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB: Memungkinkan akses ke atribut tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan akses ke atribut DynamoDB tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Persyaratan `dynamodb:Select` mencegah tindakan API dari mengembalikan atribut apapun yang tidak diperbolehkan, seperti dari proyeksi indeks. Untuk mempelajari selengkapnya tentang kunci kondisi DynamoDB, lihat Menentukan Kondisi: Menggunakan Kunci Kondisi di Panduan Pengembang Amazon DynamoDB. Untuk mempelajari tentang menggunakan beberapa ketentuan atau beberapa kunci kondisi dalam blok `Condition` dari kebijakan IAM, lihat Beberapa nilai dalam suatu ketentuan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
```

```
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select": "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB: Mengizinkan akses tingkat item ke DynamoDB berdasarkan ID Amazon Cognito

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan akses tingkat item ke tabel `MyTable` DynamoDB berdasarkan ID pengguna kumpulan identitas Amazon Cognito. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk menggunakan kebijakan ini, Anda harus menyusun tabel DynamoDB sehingga ID pengguna kumpulan identitas Amazon Cognito adalah kunci partisi. Untuk informasi selengkapnya, lihat Membuat Tabel di Panduan Pengembang Amazon DynamoDB.

Untuk mempelajari selengkapnya tentang kunci kondisi DynamoDB, lihat Menentukan Kondisi: Menggunakan Kunci Kondisi di Panduan Pengembang Amazon DynamoDB.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
```

```
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
}
```

Amazon EC2: Lampirkan atau lepaskan volume Amazon EBS ke instans EC2 berdasarkan tag

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pemilik volume EBS melampirkan atau melepaskan volume EBS mereka yang ditentukan menggunakan tag `VolumeUser` ke instans EC2 yang ditandai sebagai instance pengembangan (). `Department=Development` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk informasi selengkapnya tentang membuat kebijakan IAM untuk mengontrol akses ke sumber daya Amazon EC2, lihat Mengontrol Akses ke Sumber Daya Amazon EC2 di Panduan Pengguna Amazon EC2.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
            }
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2: Memungkinkan peluncuran instans EC2 di subnet tertentu, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan daftar informasi untuk semua objek EC2 dan meluncurkan instans EC2 di subnet tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
```

```
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Amazon EC2: Memungkinkan mengelola grup keamanan EC2 dengan pasangan nilai kunci tag tertentu secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memberikan izin kepada pengguna untuk mengambil tindakan tertentu untuk grup keamanan yang memiliki tag yang sama. Kebijakan ini memberikan izin untuk melihat grup keamanan di konsol Amazon EC2, menambah dan menghapus aturan masuk dan keluar, serta mencantumkan serta mengubah deskripsi aturan untuk grup keamanan yang ada dengan tag. `Department=Test` Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
         "ec2:ModifySecurityGroupRules",
         "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
         "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
      ],
```

```
      "Resource": [
         "arn:aws:ec2:region:111122223333:security-group/*"
      ],
      "Condition": {
         "StringEquals": {
            "aws:ResourceTag/Department": "Test"
         }
      }
     },
     {
      "Effect": "Allow",
      "Action": [
         "ec2:ModifySecurityGroupRules"
      ],
      "Resource": [
         "arn:aws:ec2:region:111122223333:security-group-rule/*"
      ]
     }
   ]
}
```

Amazon EC2: Memungkinkan memulai atau menghentikan instans EC2 yang telah ditandai pengguna, secara terprogram, dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk memulai atau menghentikan instans EC2, tetapi hanya jika tag instans `Owner` memiliki nilai nama pengguna tersebut. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

EC2: Mulai atau hentikan instans berdasarkan tanda

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan memulai atau menghentikan instance dengan pasangan nilai kunci tag`Project = DataAnalytics`, tetapi hanya berdasarkan prinsip dengan pasangan nilai kunci tag. `Department = Data` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Ketentuan dalam pengembalian kebijakan benar jika kedua bagian dari ketentuan tersebut benar. Instans harus memiliki tanda `Project=DataAnalytics`. Sebagai tambahan, prinsipal IAM (pengguna atau peran) yang membuat permintaan harus memiliki tag `Department=Data`.

> **Note**
>
> Sebagai praktik terbaik, lampirkan kebijakan dengan kunci ketentuan `aws:PrincipalTag` ke grup IAM, untuk kasus di mana beberapa pengguna mungkin memiliki tag tertentu dan beberapa mungkin tidak.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
```

```
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2: Mulai atau hentikan instans berdasarkan pada pencocokan prinsipal dan tanda sumber daya

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan prinsipal memulai atau menghentikan instans Amazon EC2 saat tag sumber daya instans dan tag prinsipal memiliki nilai yang sama untuk kunci tag. `CostCenter` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

> **Note**
>
> Sebagai praktik terbaik, lampirkan kebijakan dengan kunci ketentuan `aws:PrincipalTag` ke grup IAM, untuk kasus di mana beberapa pengguna mungkin memiliki tag tertentu dan beberapa mungkin tidak.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
```

```
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
    }
}
```

Amazon EC2: Memungkinkan akses EC2 penuh dalam Wilayah tertentu, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan akses EC2 penuh dalam Wilayah tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan. Untuk daftar kode Wilayah, lihat Wilayah yang Tersedia dalam Panduan Pengguna Amazon EC2.

Atau, Anda dapat menggunakan kunci kondisi global `aws:RequestedRegion`, yang didukung oleh semua tindakan Amazon EC2 API. Untuk informasi selengkapnya, lihat Contoh: Membatasi akses ke Wilayah tertentu di Panduan Pengguna Amazon EC2.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Amazon EC2: Memungkinkan memulai atau menghentikan instans EC2 dan memodifikasi grup keamanan, secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan memulai atau menghentikan instans EC2 tertentu dan memodifikasi grup keamanan tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks*

placeholder yang dicetak miring dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Amazon EC2: Memerlukan MFA (GetSessionToken) untuk operasi EC2 tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan akses penuh ke semua operasi AWS API di Amazon EC2. Namun, itu secara tegas menolak akses ke `StopInstances` dan `TerminateInstances` Operasi API jika pengguna tidak diautentikasi menggunakan Multi-Factor Authentication (MFA). Untuk melakukan ini secara terprogram, pengguna harus menyertakan nilai opsional `SerialNumber` dan `TokenCode` saat memanggil operasi `GetSessionToken`. Operasi ini mengembalikan kredensial sementara yang diautentikasi

menggunakan MFA. Untuk mempelajari lebih lanjut tentang GetSessionToken, lihatMeminta kredensi untuk pengguna di lingkungan yang tidak tepercaya.

Apa yang dikerjakan oleh kebijakan ini?

- Pernyataan `AllowAllActionsForEC2` mengizinkan semua tindakan Amazon EC2
- Pernyataan `DenyStopAndTerminateWhenMFAIsNotPresent` menolak tindakan `StopInstances` dan `TerminateInstances` saat konteks MFA hilang. Ini berarti tindakan tersebut ditolak saat konteks multi-factor authentication hilang (artinya MFA tidak digunakan). Penolakan mennimpa izin.

> **Note**
>
> Pemeriksaaan ketentuan untuk `MultiFactorAuthPresent` dalam pernyataan `Deny` tidak boleh berupa `{"Bool":{"aws:MultiFactorAuthPresent":false}}` karena kunci tersebut tidak ada dan tidak dapat dievaluasi saat MFA tidak digunakan. Sebagai gantinya, gunakan periksa `BoolIfExists` untuk melihat apakah kuncinya ada sebelum memeriksa nilai. Untuk informasi selengkapnya, lihat ... IfExists operator kondisi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
```

```
        }
    ]
}
```

Amazon EC2: Membatasi penghentian instans EC2 ke rentang alamat IP

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang membatasi instans EC2 dengan mengizinkan tindakan, tetapi secara eksplisit menolak akses ketika permintaan berasal dari luar rentang IP yang ditentukan. Kebijakan ini berguna saat alamat IP untuk perusahaan Anda berada dalam cakupan tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Jika kebijakan ini digunakan dalam kombinasi dengan kebijakan lain yang memungkinkan `ec2:TerminateInstances` tindakan (seperti kebijakan FullAccess AWS terkelola AmazonEC2), maka akses ditolak. Hal ini dikarenakan pernyataan penolakan yang eksplisit lebih diutamakan daripada pernyataan yang membolehkan. Untuk informasi selengkapnya, lihat the section called "Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun.".

> **Important**
>
> Kunci `aws:SourceIp` kondisi menolak akses ke AWS layanan, seperti AWS CloudFormation, yang membuat panggilan atas nama Anda. Untuk informasi selengkapnya tentang penggunaan kunci kondisi `aws:SourceIp`, lihat AWS kunci konteks kondisi global.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
```

```
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM: Akses API simulator kebijakan

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan API simulator kebijakan untuk kebijakan yang dilampirkan ke pengguna, grup, atau peran saat ini. Akun AWS Kebijakan ini juga mengizinkan akses untuk menyimulasikan kebijakan yang kurang sensitif untuk diteruskan ke API sebagai strings. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Untuk mengizinkan pengguna mengakses konsol simulator kebijakan guna mensimulasikan kebijakan yang dilampirkan ke pengguna, grup, atau peran saat ini Akun AWS, lihatIAM: Akses konsol simulator kebijakan.

IAM: Akses konsol simulator kebijakan

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan konsol simulator kebijakan untuk kebijakan yang dilampirkan ke pengguna, grup, atau peran saat ini. Akun AWS Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

Anda dapat mengakses konsol IAM Policy Simulator di: https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetGroup",
                "iam:GetGroupPolicy",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

IAM: Asumsikan peran yang memiliki tag tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk mengambil peran dengan pasangan nilai kunci tag. `Project = ExampleCorpABC` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Jika ada peran dengan tag ini pada akun yang sama dengan pengguna, maka pengguna dapat mengambil peran tersebut. Jika ada peran dengan tanda ini pada akun selain milik pengguna, peran tersebut memerlukan izin tambahan. Kebijakan kepercayaan peran lintas-akun juga harus mengizinkan pengguna atau semua anggota dari akun pengguna untuk mengambil peran itu. Untuk informasi tentang penggunaan peran untuk akses lintas-akun, lihat Akses untuk IAM pengguna lain Akun AWS yang Anda miliki.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM: Mengizinkan dan menolak akses ke beberapa layanan secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan akses penuh ke beberapa layanan dan akses pengelolaan mandiri terbatas di IAM. Ia juga menolak akses ke bucket `logs` Amazon S3 atau contoh `i-1234567890abcdef0` Amazon EC2. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

> **Warning**
>
> Kebijakan ini mengizinkan akses penuh ke setiap tindakan dan sumber daya dalam beberapa layanan. Kebijakan ini harus diterapkan hanya bagi administrator yang tepercaya.

Anda dapat menggunakan kebijakan ini sebagai batasan izin untuk menentukan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada pengguna IAM. Untuk informasi selengkapnya, lihat Mendelegasikan tanggung jawab kepada orang lain menggunakan batas izin. Ketika kebijakan itu digunakan sebagai batasan izin bagi pengguna, pernyataan itu menentukan batasan berikut:

- Pernyataan `AllowServices` mengizinkan akses penuh ke layanan AWS tertentu. Artinya, tindakan pengguna dalam layanan ini hanya dibatasi oleh kebijakan izin yang melekat pada pengguna.
- Pernyataan `AllowIAMConsoleForCredentials` mengizinkan akses untuk mencantumkan semua pengguna IAM. Akses ini diperlukan untuk menavigasi halaman Pengguna di AWS Management Console. Ia juga mengizinkan untuk melihat persyaratan kata sandi untuk akun, yang diperlukan bagi pengguna untuk mengubah kata sandinya sendiri.
- Pernyataan `AllowManageOwnPasswordAndAccessKeys` ini mengizinkan pengguna mengelola kata sandi konsol dan kunci akses programnya sendiri. Ini penting karena jika kebijakan lain memberi akses IAM penuh ke pengguna, pengguna tersebut lalu dapat mengubah izinnya sendiri atau izin pengguna lain. Pernyataan ini mencegah hal tersebut terjadi.
- Pernyataan `DenyS3Logs` itu secara eksplisit menolak akses ke `logs` bucket. Kebijakan ini menerapkan batasan perusahaan kepada pengguna.
- Pernyataan `DenyEC2Production` secara eksplisit menolak akses ke instans `i-1234567890abcdef0`.

Kebijakan ini tidak mengizinkan akses ke layanan atau tindakan lain. Jika kebijakan digunakan sebagai batas izin pada pengguna, meskipun kebijakan lain yang dilampirkan pada pengguna mengizinkan tindakan tersebut, AWS menolak permintaan tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
```

```
}
```

IAM: Tambahkan tag tertentu ke pengguna dengan tag tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penambahan kunci tag `Department` dengan nilai tag `MarketingDevelopment`, atau `QualityAssurance` ke pengguna IAM. Pengguna tersebut harus sudah menyertakan pasangan kunci tag — nilai. `JobFunction = manager` Anda dapat menggunakan kebijakan ini untuk mewajibkan satu manajer hanya dimiliki oleh satu dari tiga departemen. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Pernyataan `ListTagsForAllUsers` memungkinkan melihat tanda untuk semua pengguna di akun Anda.

Kondisi pertama dalam `TagManagerWithSpecificDepartment` pernyataan menggunakan `StringEquals` operator kondisi. Kondisi akan kembali sebagai benar bila kedua bagian dari kondisi adalah benar. Pengguna yang akan ditandai harus sudah memiliki tanda `JobFunction=Manager`. Permintaan itu harus menyertakan kunci tanda `Department` dengan satu nilai tanda yang terdaftar.

Kondisi kedua menggunakan operator kondisi `ForAllValues:StringEquals`. Kondisi mengembalikan benar jika semua kunci tanda dalam permintaan sesuai dengan kunci dalam kebijakan. Artinya satu-satunya kunci tanda dalam permintaan tersebut harus `Department`. Untuk informasi selengkapnya tentang penggunaan `ForAllValues`, lihat Kunci konteks multivaluasi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
```

```
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM: Tambahkan tag tertentu dengan nilai tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan hanya menambahkan kunci tag `CostCenter` dan nilai tag `A-123` atau nilai tag `B-456` ke pengguna atau peran IAM mana pun. Anda dapat menggunakan kebijakan ini untuk membatasi penandaan ke kunci tanda tertentu dan serangkaian nilai tanda. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Pernyataan `ConsoleDisplay` memungkinkan melihat tanda untuk semua pengguna dan peran dalam akun Anda.

Kondisi pertama dalam `AddTag` pernyataan menggunakan `StringEquals` operator kondisi. Kondisi mengembalikan benar jika permintaan memasukkan kunci tanda `CostCenter` dengan satu nilai tanda yang terdaftar.

Kondisi kedua menggunakan operator kondisi `ForAllValues:StringEquals`. Kondisi mengembalikan benar jika semua kunci tanda dalam permintaan sesuai dengan kunci dalam kebijakan. Artinya satu-satunya kunci tanda dalam permintaan tersebut harus `CostCenter`. Untuk informasi selengkapnya tentang penggunaan `ForAllValues`, lihat Kunci konteks multivaluasi.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM: Buat pengguna baru hanya dengan tag tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pembuatan pengguna IAM tetapi hanya dengan satu atau kedua kunci dan tag. `Department` `JobFunction` Kunci tanda `Department` harus memiliki nilai tanda `Development` atau `QualityAssurance`. Kunci tanda `JobFunction` harus memiliki nilia tanda `Employee`. Anda dapat menggunakan kebijakan ini untuk mewajibkan bahwa pengguna baru memiliki fungsi dan

departemen tugas tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kondisi pertama dalam pernyataan itu menggunakan `StringEqualsIfExists` operator kondisi. Jika tanda dengan kunci `Department` atau `JobFunction` ada dalam permintaan, tanda tersebut harus memiliki nilai yang ditentukan. Jika tidak ada kunci yang muncul, maka kondisi ini akan dievaluasi sebagai benar. Satu-satunya cara bahwa kondisi itu dievaluasi sebagai salah adalah bila satu dari kunci kondisi yang ditentukan muncul dalam permintaan, tetapi memiliki nilai berbeda dari yang diizinkan. Untuk informasi selengkapnya tentang penggunaan `IfExists`, lihat ... IfExists operator kondisi.

Kondisi kedua menggunakan operator kondisi `ForAllValues:StringEquals`. Kondisi akan mengembalikan benar bila ada kecocokan di antara setiap kunci tanda khusus yang ditentukan dalam permintaan, dan setidaknya satu nilai dalam kebijakan. Artinya, semua tanda dalam permintaan harus ada dalam daftar ini. Namun, permintaan tersebut hanya dapat memasukkan satu tanda ke dalam daftar. Misalnya, Anda dapat membuat pengguna IAM hanya dengan `Department=QualityAssurance` tag. Namun, Anda tidak dapat membuat pengguna IAM dengan `JobFunction=employee` tag dan `Project=core` tag. Untuk informasi selengkapnya tentang penggunaan `ForAllValues`, lihat Kunci konteks multivaluasi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
```

```
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM: Menghasilkan dan mengambil laporan kredensi IAM

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna membuat dan mengunduh laporan yang mencantumkan semua pengguna IAM di dalamnya. Akun AWS Laporan ini termasuk status dari kredensial pengguna, termasuk kata sandi, access key, perangkat MFA, dan sertifikat tanda tangan. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

Untuk informasi selengkapnya tentang laporan kredensial, lihat Hasilkan laporan kredensi untuk Anda Akun AWS.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:GetCredentialReport"
        ],
        "Resource": "*"
    }
}
```

IAM: Memungkinkan mengelola keanggotaan grup secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan memperbarui keanggotaan grup yang dipanggil. `MarketingTeam` Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Apa yang dikerjakan oleh kebijakan ini?

- Pernyataan `ViewGroups` memungkinkan penggunanya mencantumkan semua pengguna dan grup di AWS Management Console. Ia juga memungkinkan pengguna untuk melihat informasi dasar tentang pengguna di akun. Izin ini harus ada di dalam pernyataannya sendiri karena mereka tidak mendukung atau tidak perlu menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`.
- Pernyataan `ViewEditThisGroup` ini memungkinkan pengguna melihat informasi tentang `MarketingTeam` grup, dan menambahkan dan menghapus pengguna dari grup tersebut.

Kebijakan ini tidak mengizinkan pengguna untuk melihat atau mengedit izin dari pengguna atau `MarketingTeam` dari grup.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:ListGroupsForUser"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM: Kelola tag tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penambahan dan penghapusan tag IAM dengan kunci tag `Department` dari entitas IAM (pengguna dan peran). Kebijakan ini tidak membatasi nilai `Department` tag. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM: Lulus peran IAM ke layanan tertentu AWS

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan meneruskan peran layanan IAM apa pun ke layanan Amazon. CloudWatch Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Peran layanan adalah peran IAM yang menentukan AWS layanan sebagai prinsipal yang dapat mengambil peran tersebut. Ini memungkinkan layanan untuk mengambil peran dan mengakses sumber daya di layanan lain atas nama Anda. Untuk memungkinkan Amazon CloudWatch mengambil peran yang Anda lewati, Anda harus menentukan prinsip `cloudwatch.amazonaws.com` layanan sebagai prinsipal dalam kebijakan kepercayaan peran Anda. Prinsipiel layanan ditentukan oleh layanan tersebut. Untuk mempelajari prinsipal layanan dari layanan, lihat dokumentasi untuk

layanan tersebut. Untuk beberapa layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang Ya dalam kolom Peran Terkait-Layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut. Cari `amazonaws.com` untuk melihat prinsipal layanan.

Untuk mempelajari penerusan peran layanan ke layanan, lihat Berikan izin pengguna untuk meneruskan peran ke layanan AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:PassedToService": "cloudwatch.amazonaws.com"}
            }
        }
    ]
}
```

IAM: Mengizinkan akses hanya-baca ke konsol IAM tanpa melaporkan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk melakukan tindakan IAM apa pun yang dimulai dengan string atau. `Get` `List` Saat pengguna bekerja dengan konsol, konsol membuat permintaan ke IAM untuk mencantumkan grup, pengguna, peran, dan kebijakan, dan untuk menghasilkan laporan tentang sumber daya tersebut.

Tanda bintang berfungsi sebagai wildcard. Saat Anda menggunakan `iam:Get*` kebijakan, izin yang dihasilkan mencakup semua tindakan IAM yang dimulai`Get`, seperti `GetUser` dan. `GetRole` Wildcard berguna jika jenis entitas baru ditambahkan ke IAM di masa mendatang. Dalam hal itu, izin yang diberikan oleh kebijakan secara otomatis memungkinkan pengguna mendaftar dan mendapat rincian mengenai entitas baru tersebut.

Kebijakan ini tidak dapat digunakan untuk membuat laporan atau detail layanan yang terakhir diakses. Untuk kebijakan yang berbeda yang mengizinkan ini, lihat IAM: Mengizinkan akses hanya-baca ke konsol IAM.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
}
```

IAM: Mengizinkan akses hanya-baca ke konsol IAM

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM melakukan tindakan IAM apa pun yang dimulai dengan string,, atau. `Get` `List` `Generate` Saat pengguna bekerja dengan konsol IAM, konsol membuat permintaan untuk mencantumkan grup, pengguna, peran, dan kebijakan, dan untuk menghasilkan laporan tentang sumber daya tersebut.

Tanda bintang berfungsi sebagai wildcard. Saat Anda menggunakan `iam:Get*` kebijakan, izin yang dihasilkan mencakup semua tindakan IAM yang dimulai`Get`, seperti `GetUser` dan. `GetRole` Menggunakan wildcard bermanfaat, terutama jika jenis entitas baru ditambahkan ke IAM di masa depan. Dalam hal itu, izin yang diberikan oleh kebijakan secara otomatis memungkinkan pengguna mendaftar dan mendapat rincian mengenai entitas baru tersebut.

Gunakan kebijakan ini untuk akses konsol yang menyertakan izin untuk menghasilkan laporan atau detail layanan yang terakhir diakses. Untuk kebijakan yang berbeda yang tidak memungkinkan pembuatan tindakan, lihat IAM: Mengizinkan akses hanya-baca ke konsol IAM tanpa melaporkan.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*",
            "iam:Generate*"
        ],
        "Resource": "*"
    }
}
```

IAM: Memungkinkan pengguna IAM tertentu untuk mengelola grup secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM tertentu mengelola grup. `AllUsers` Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Apa yang dikerjakan oleh kebijakan ini?

- Pernyataan `AllowAllUsersToListAllGroups` ini memungkinkan pendaftaran semua grup. Hal ini diperlukan untuk akses konsol. Izin ini harus berada dalam pernyataannya sendiri karena tidak mendukung ARN sumber daya. Alih-alih, izin ini menentukan `"Resource" : "*"`.
- Pernyataan `AllowAllUsersToViewAndManageThisGroup` ini mengizinkan semua tindakan grup yang dapat dilakukan pada tipe sumber daya grup. Ia tidak mengizinkan `ListGroupsForUser` tindakan, yang dapat dilakukan pada tipe sumber daya pengguna dan bukan tipe sumber daya grup. Untuk informasi selengkapnya tentang jenis sumber daya yang dapat Anda tentukan untuk tindakan IAM, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Identity and Access Management.
- Pernyataan `LimitGroupManagementAccessToSpecificUsers` itu menolak pengguna dengan nama tertentu mengakses untuk menulis dan pengelolaan izin tindakan grup. Ketika pengguna yang ditentukan dalam kebijakan mencoba untuk membuat perubahan pada grup, pernyataan ini tidak menolak permintaan tersebut. Permintaan tersebut akan diizinkan oleh `AllowAllUsersToViewAndManageThisGroup` pernyataan. Jika pengguna lain mencoba melakukan operasi ini, permintaannya ditolak. Anda dapat melihat tindakan IAM yang dijelaskan dengan Tulis atau pengelolaan izin tingkat akses sambil membuat kebijakan ini di konsol IAM. Untuk melakukan hal ini, beralihlah dari tab JSON ke tabeditor Visual. Untuk informasi selengkapnya tentang tingkat akses. lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Identity and Access Management.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
```

```
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
                        "srodriguez",
                        "mjackson",
                        "adesai"
                    ]
                }
            }
        }
    ]
}
```

IAM: Memungkinkan pengaturan persyaratan kata sandi akun secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna untuk melihat dan memperbarui persyaratan kata sandi akun mereka. Persyaratan kata sandi tersebut menentukan kompleksitas persyaratan dan periode rotasi wajib

untuk kata sandi para anggota akun. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

Untuk mempelajari bagaimana mengatur persyaratan kata sandi akun untuk akun Anda, lihat Menetapkan kebijakan kata sandi akun untuk IAM pengguna.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:UpdateAccountPasswordPolicy"
        ],
        "Resource": "*"
    }
}
```

IAM: Akses API simulator kebijakan berdasarkan jalur pengguna

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan penggunaan API simulator kebijakan hanya untuk pengguna yang memiliki jalur. `Department/Development` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

> **Note**
>
> Untuk membuat kebijakan yang memungkinkan penggunaan konsol simulator kebijakan untuk pengguna yang memiliki jalur `Department/Development`, lihat IAM: Akses konsol simulator kebijakan berdasarkan jalur pengguna.

IAM: Akses konsol simulator kebijakan berdasarkan jalur pengguna

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan konsol simulator kebijakan hanya untuk pengguna yang memiliki jalur tersebut. `Department/Development` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Anda dapat mengakses konsol Simulator Kebijakan IAM di: https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
```

```
}
```

IAM: Memungkinkan pengguna IAM untuk mengelola sendiri perangkat MFA

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengelola sendiri perangkat autentikasi multi-faktor (MFA) mereka. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

> **Note**
>
> Jika pengguna IAM dengan kebijakan ini tidak diautentikasi oleh MFA, kebijakan ini menolak akses ke semua AWS tindakan kecuali yang diperlukan untuk mengautentikasi menggunakan MFA. Jika Anda menambahkan izin ini untuk pengguna yang masuk AWS, mereka mungkin perlu keluar dan masuk kembali untuk melihat perubahan ini.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

IAM: Memungkinkan pengguna IAM memperbarui kredensialnya sendiri secara terprogram dan di konsol

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM memperbarui kunci akses mereka sendiri, menandatangani sertifikat, kredensi khusus layanan, dan kata sandi. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

Untuk mempelajari bagaimana pengguna dapat mengubah kata sandinya sendiri di konsol, lihat the section called “Bagaimana pengguna IAM mengubah kata sandi mereka sendiri”.

IAM: Melihat informasi layanan yang terakhir diakses untuk kebijakan Organizations

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan layanan melihat informasi yang terakhir diakses untuk kebijakan Organizations tertentu. Kebijakan ini memungkinkan pengambilan data untuk kebijakan kontrol layanan (SCP) dengan `p-policy123` ID. Orang yang membuat dan melihat laporan harus diautentikasi

menggunakan kredensi akun AWS Organizations manajemen. Kebijakan ini memungkinkan pemohon mengambil data bagi entitas Organisasi di dalam organisasinya. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *italicized placeholder text* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Untuk informasi penting tentang informasi yang diakses terakhir, termasuk izin yang diperlukan, pemecahan masalah, dan Wilayah yang didukung, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
            }
        }
    ]
}
```

IAM: Membatasi kebijakan terkelola yang dapat diterapkan pada pengguna, grup, atau peran IAM

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang membatasi kebijakan yang dikelola dan AWS dikelola pelanggan yang dapat diterapkan pada pengguna, grup, atau peran IAM. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram

dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
        }
    }
}
```

AWS: Tolak akses ke sumber daya di luar akun Anda kecuali kebijakan IAM yang AWS dikelola

Menggunakan `aws:ResourceAccount` kebijakan berbasis identitas Anda dapat memengaruhi pengguna atau kemampuan peran untuk memanfaatkan beberapa layanan yang memerlukan interaksi dengan sumber daya di akun yang dimiliki oleh layanan.

Anda dapat membuat kebijakan dengan pengecualian untuk mengizinkan kebijakan IAM AWS terkelola. Akun yang dikelola layanan di luar AWS Organizations Anda memiliki Kebijakan IAM Terkelola. Ada empat tindakan IAM yang mencantumkan dan mengambil kebijakan AWS-managed. Gunakan tindakan ini dalam `NotAction`elemen pernyataan. `AllowAccessToS3ResourcesInSpecificAccountsAndSpecificService1`dalam kebijakan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda: Mengizinkan fungsi Lambda mengakses tabel Amazon DynamoDB

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan akses baca dan tulis ke tabel Amazon DynamoDB tertentu. Kebijakan ini juga memungkinkan penulisan file log ke CloudWatch Log. Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Untuk menggunakan kebijakan ini, lampirkan kebijakan tersebut pada peran layanan Lambda. Peran layanan adalah peran yang Anda buat di akun Anda untuk mengizinkan layanan melakukan tindakan atas nama Anda. Peran layanan itu harus dimasukkan AWS Lambda sebagai kepala sekolah dalam kebijakan kepercayaan. Untuk detail tentang cara menggunakan kebijakan ini, lihat Cara Membuat Kebijakan AWS IAM untuk Memberikan AWS Lambda Akses ke Tabel AWS Amazon DynamoDB di Blog Keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
```

```
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
            "Sid": "WriteLogStreamsAndGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: Memungkinkan akses database RDS penuh dalam Wilayah tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan akses database RDS penuh dalam Wilayah tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:region:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}
```

Amazon RDS: Memungkinkan memulihkan database RDS, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pemulihan database RDS. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: Memungkinkan pemilik tag akses penuh ke sumber daya RDS yang telah mereka tag

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pemilik tag akses penuh ke sumber daya RDS yang telah mereka tag. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "rds:DeleteDBInstance",
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
```

```
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
```

```
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Amazon S3: Memungkinkan pengguna Amazon Cognito mengakses objek di bucket mereka

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna Amazon Cognito mengakses objek di bucket S3 tertentu. Kebijakan ini memungkinkan akses hanya ke obyek dengan nama yang mencantumkan `cognito`, nama aplikasi, dan ID pengguna federasi, yang diwakili oleh variabel ${cognito-identity.amazonaws.com:sub} Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *`teks placeholder yang dicetak miring`* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

> **Note**
>
> Nilai 'sub' yang digunakan dalam kunci obyek bukanlah nilai sub pengguna di Kumpulan Pengguna, namun identitas id yang diasosiasikan dengan pengguna di Kumpulan Identitas.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
      }
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

Amazon Cognito menyediakan autentikasi, otorisasi, dan pengelolaan pengguna untuk aplikasi web dan seluler Anda. Pengguna Anda bisa langsung masuk dengan nama pengguna dan kata sandi, atau lewat pihak ketiga seperti Facebook, Amazon, atau Google.

Dua komponen utama Amazon Cognito adalah kumpulan pengguna dan kumpulan identitas. Kumpulan pengguna adalah direktori pengguna yang menyediakan opsi pendaftaran dan masuk bagi pengguna aplikasi Anda. Kumpulan identitas memungkinkan Anda memberi pengguna akses ke

AWS layanan lain. Anda dapat menggunakan kolam identitas dan kolam pengguna secara terpisah atau bersama-sama.

Untuk informasi selengkapnya tentang Amazon Cognito, lihat Panduan Pengguna Amazon Cognito.

Amazon S3: Memungkinkan pengguna federasi mengakses ke direktori home S3 mereka, secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna federasi mengakses objek bucket direktori home mereka sendiri di S3. Direktori rumah adalah bucket yang mencakup home folder dan folders untuk pengguna federasi individual. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`italicized placeholder text`* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Variabel `${aws:userid}` dalam kebijakan ini memutuskan untuk `role-id:specified-name`. Bagian `role-id` ID pengguna gabungan adalah pengidentifikasi unik yang diberikan ke peran pengguna gabungan selama pembuatan. Untuk informasi selengkapnya, lihat Pengidentifikasi unik. `specified-name`Ini adalah RoleSessionName parameter yang diteruskan ke `AssumeRoleWithWebIdentity` permintaan ketika pengguna federasi mengambil peran mereka.

Anda dapat melihat ID peran menggunakan AWS CLI perintah`aws iam get-role --role-name` *`specified-name`*. Misalnya, bayangkan Anda menentukan nama ramah `John` dan CLI mengembalikan ID peran`AROAXXT2NJT7D3SIQN7Z6`. Dalam kasus ini, ID pengguna gabungan adalah `AROAXXT2NJT7D3SIQN7Z6:John`. Kebijakan ini kemudian memungkinkan pengguna federasi John untuk mengakses bucket Amazon S3 dengan awalan. `AROAXXT2NJT7D3SIQN7Z6:John`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
```

```
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3: Akses Bucket S3, tetapi bucket produksi ditolak tanpa baru-baru ini MFA

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan administrator Amazon S3 mengakses bucket apa pun, termasuk memperbarui, menambahkan, dan menghapus objek. Namun, secara eksplisit menolak akses ke `amzn-s3-demo-bucket-production` bucket jika pengguna belum masuk menggunakan otentikasi multi-faktor (MFA) dalam tiga puluh menit terakhir. Kebijakan ini memberikan izin yang diperlukan untuk melakukan tindakan ini di konsol atau secara terprogram menggunakan AWS CLI atau AWS API. Untuk menggunakan kebijakan ini, ganti *`italicized placeholder text`* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kebijakan ini tidak pernah mengizinkan akses terprogram ke bucket `amzn-s3-demo-bucket` menggunakan access key pengguna. Hal ini dicapai menggunakan kunci kondisi `aws:MultiFactorAuthAge` dengan operator kondisi `NumericGreaterThanIfExists`. Kondisi kebijakan ini kembali `true` jika MFA tidak ada atau jika usia MFA lebih dari 30 menit. Dalam situasi tersebut, akses ditolak. Untuk mengakses `amzn-s3-demo-bucket-production` bucket secara terprogram, administrator S3 harus menggunakan kredenal sementara yang dihasilkan dalam 30 menit terakhir menggunakan operasi. GetSessionTokenAPI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
```

```
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-production/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3: Memungkinkan IAM pengguna mengakses direktori home S3 mereka, secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan IAM pengguna mengakses objek bucket direktori home mereka sendiri di S3. Direktori rumah adalah bucket yang mencakup home folder dan folders untuk pengguna individual. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *italicized placeholder text* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kebijakan ini tidak akan berfungsi saat menggunakan IAM peran karena `aws:username` variabel tidak tersedia saat menggunakan IAM peran. Untuk detail tentang nilai kunci prinsipal, lihat Nilai-nilai kunci utama.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
```

```
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3: Batasi manajemen ke bucket S3 tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang membatasi pengelolaan bucket Amazon S3 ke bucket tertentu. Kebijakan ini memberikan izin untuk melakukan semua tindakan Amazon S3, tetapi menolak akses ke Layanan AWS setiap kecuali Amazon S3. Lihat contoh berikut ini. Menurut kebijakan ini, Anda hanya dapat mengakses tindakan Amazon S3 yang dapat Anda lakukan pada bucket S3 atau sumber daya objek S3. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Jika kebijakan ini digunakan dalam kombinasi dengan kebijakan lain (seperti kebijakan FullAccess AWS terkelola AmazonS3 FullAccess atau AmazonEC2) yang memungkinkan tindakan ditolak oleh kebijakan ini, maka akses ditolak. Hal ini dikarenakan pernyataan penolakan yang eksplisit

lebih diutamakan daripada pernyataan yang membolehkan. Untuk informasi selengkapnya, lihat the section called “Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun.”.

> **Warning**
>
> `NotAction` dan `NotResource` adalah elemen kebijakan lanjutan yang harus digunakan dengan seksama. Kebijakan ini menolak akses ke setiap AWS layanan kecuali Amazon S3. Jika Anda melampirkan kebijakan ini kepada pengguna, kebijakan lain yang mengizinkan layanan lain diabaikan dan akses ditolak.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

Berikan akses baca dan tulis ke objek bucket Amazon S3

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang mengizinkan `Read` dan `Write` mengakses objek di bucket Amazon S3 tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari atau. AWS API AWS CLI Untuk menggunakan kebijakan ini, ganti *`italicized placeholder text`* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Tindakan `s3:*Object` ini menggunakan wildcard sebagai bagian dari nama tindakan. Pernyataan ini `AllObjectActions` mengizinkan `GetObject`, `DeleteObject`, `PutObject`, dan tindakan Amazon S3 lainnya yang berakhir dengan kata “Obyek”.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

> **Note**
>
> Untuk mengizinkan `Read` dan `Write` akses ke objek dalam bucket Amazon S3 dan juga termasuk izin tambahan untuk akses konsol, lihat Amazon S3: Memungkinkan akses baca dan tulis ke objek di Bucket S3, secara terprogram dan di konsol.

Amazon S3: Memungkinkan akses baca dan tulis ke objek di Bucket S3, secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang mengizinkan `Read` dan `Write` mengakses objek dalam bucket S3 tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *`italicized placeholder text`* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Tindakan `s3:*Object` ini menggunakan wildcard sebagai bagian dari nama tindakan. Pernyataan ini `AllObjectActions` mengizinkan `GetObject`, `DeleteObject`, `PutObject`, dan tindakan Amazon S3 lainnya yang berakhir dengan kata “Obyek”.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
        }
    ]
}
```

Kelola IAM kebijakan

IAMmemberi Anda alat untuk membuat dan mengelola semua jenis IAM kebijakan (kebijakan terkelola dan kebijakan sebaris). Untuk menambahkan izin ke IAM identitas (IAMpengguna, grup, atau peran), Anda membuat kebijakan, memvalidasi kebijakan, dan kemudian melampirkan kebijakan

ke identitas. Anda dapat melampirkan beberapa kebijakan ke satu identitas, dan setiap kebijakan dapat berisi beberapa izin.

Topik

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang AWS kebijakan.

- Untuk informasi selengkapnya tentang berbagai jenis IAM kebijakan, lihatKebijakan dan izin di AWS Identity and Access Management.
- Untuk informasi umum tentang penggunaan kebijakan dalamIAM, lihatManajemen akses untuk AWS sumber daya.
- Untuk informasi tentang cara menggunakan IAM Access Analyzer untuk membuat IAM kebijakan yang didasarkan pada aktivitas akses untuk entitas, lihat Pembuatan kebijakan IAM Access Analyzer.
- Untuk informasi tentang bagaimana izin dievaluasi ketika beberapa kebijakan diberlakukan untuk IAM identitas tertentu, lihat. Logika evaluasi kebijakan
- Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Tentukan IAM izin khusus dengan kebijakan terkelola pelanggan

Kebijakan menentukan izin untuk identitas atau sumber daya di. AWS Anda dapat membuat kebijakan yang dikelola pelanggan dalam IAM menggunakan AWS Management Console, AWS CLI,

atau AWS API. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiri Akun AWS. Anda kemudian dapat melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda Akun AWS.

Kebijakan berbasis identitas adalah kebijakan yang melekat pada identitas di. IAM Kebijakan berbasis identitas dapat mencakup kebijakan terkelola, kebijakan yang AWS dikelola pelanggan, dan kebijakan inline. AWS kebijakan terkelola dibuat dan dikelola oleh AWS, dan Anda dapat menggunakannya tetapi tidak mengelolanya. Kebijakan inline adalah kebijakan yang Anda buat dan sematkan langsung ke grup IAM pengguna, pengguna, atau peran. Kebijakan inline tidak dapat digunakan kembali pada identitas lain atau dikelola di luar identitas di mana mereka ada. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Umumnya lebih baik menggunakan kebijakan yang dikelola pelanggan daripada kebijakan inline atau kebijakan AWS terkelola. AWS kebijakan terkelola biasanya memberikan izin administratif atau hanya-baca yang luas. Untuk keamanan terbesar, berikan hak istimewa paling sedikit, yang berarti hanya memberikan izin yang diperlukan untuk melakukan tugas pekerjaan tertentu.

Ketika Anda membuat atau mengedit IAM kebijakan, secara otomatis AWS dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif dengan sedikit hak istimewa dalam pikiran. Dalam AWS Management Console, IAM mengidentifikasi kesalahan JSON sintaks, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat Validasi kebijakan IAMAccess Analyzer.

Anda dapat menggunakan AWS Management Console, AWS CLI, atau AWS API untuk membuat kebijakan yang dikelola pelanggan diIAM. Untuk informasi selengkapnya tentang menggunakan AWS CloudFormation templat untuk menambah atau memperbarui kebijakan, lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan AWS CloudFormation Pengguna.

Topik

- Buat IAM kebijakan (konsol)
- Buat IAM kebijakan (AWS CLI)
- Buat IAM kebijakan (AWS API)

Buat IAM kebijakan (konsol)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS Management Console untuk membuat kebijakan yang dikelola pelanggan diIAM. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiri. Akun AWS Anda kemudian dapat melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda Akun AWS.

Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Topik

- Membuat IAM kebijakan
- Membuat kebijakan menggunakan JSON editor
- Membuat kebijakan dengan editor visual
- Mengimpor kebijakan terkelola yang ada

Membuat IAM kebijakan

Anda dapat membuat kebijakan terkelola pelanggan AWS Management Console menggunakan salah satu metode berikut:

- JSON— Tempel dan sesuaikan contoh kebijakan berbasis identitas yang dipublikasikan.
- Editor visual — Susun kebijakan baru dari nol di editor visual. Jika Anda menggunakan editor visual, Anda tidak perlu memahami JSON sintaks.
- Impor — Impor dan sesuaikan kebijakan terkelola dari akun Anda. Anda dapat mengimpor kebijakan AWS terkelola atau kebijakan terkelola pelanggan yang sebelumnya Anda buat.

Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Membuat kebijakan menggunakan JSON editor

Anda dapat mengetik atau menempelkan kebijakan JSON dengan memilih JSONopsi. Metode ini berguna untuk menyalin contoh kebijakan untuk dipakai di akun Anda. Atau, Anda dapat mengetik dokumen JSON kebijakan Anda sendiri di JSON editor. Anda juga dapat menggunakan JSONopsi untuk beralih antara editor visual dan JSON membandingkan tampilan.

Saat Anda membuat atau mengedit kebijakan di JSON editor, IAM lakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAMmengidentifikasi kesalahan JSON sintaks, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda menyempurnakan kebijakan lebih lanjut.

Dokumen JSON kebijakan terdiri dari satu atau lebih pernyataan. Setiap pernyataan harus berisi semua tindakan yang berbagi efek yang sama (`Allow` atau `Deny`) dan mendukung sumber daya dan kondisi yang sama. Jika satu tindakan mengharuskan Anda menentukan semua resource (`"*"`) dan tindakan lain mendukung Amazon Resource Name (ARN) sumber daya tertentu, tindakan tersebut harus ada dalam dua JSON pernyataan terpisah. Untuk detail tentang ARN format, lihat Amazon Resource Name (ARN) di Referensi Umum AWS Panduan. Untuk informasi umum tentang IAM kebijakan, lihatKebijakan dan izin di AWS Identity and Access Management. Untuk informasi tentang bahasa IAM kebijakan, lihatIAMJSONreferensi kebijakan.

Untuk menggunakan editor JSON kebijakan untuk membuat kebijakan

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih JSONopsi.
5. Ketik atau tempel dokumen JSON kebijakan. Untuk detail tentang bahasa IAM kebijakan, lihatIAMJSONreferensi kebijakan.
6. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

   > **Note**
   >
   > Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

7. (Opsional) Saat Anda membuat atau mengedit kebijakan di AWS Management Console, Anda dapat membuat JSON atau templat YAML kebijakan yang dapat Anda gunakan dalam AWS CloudFormation templat.

Untuk melakukan ini, di editor Kebijakan pilih Tindakan, lalu pilih Buat CloudFormation templat. Untuk mempelajari selengkapnya, AWS CloudFormation lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan AWS CloudFormation Pengguna.

8. Setelah selesai menambahkan izin ke kebijakan, pilih Berikutnya.
9. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda.
10. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
11. Pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.

Setelah Anda membuat kebijakan, Anda dapat melampirkannya ke grup, pengguna, atau peran Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Membuat kebijakan dengan editor visual

Editor visual di IAM konsol memandu Anda membuat kebijakan tanpa harus menulis JSON sintaks. Untuk melihat contoh penggunaan editor visual dalam pembuatan kebijakan, lihat the section called “Mengontrol akses ke identitas”.

Untuk menggunakan editor visual dalam pembuatan kebijakan

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, temukan bagian Pilih layanan, lalu pilih AWS layanan. Anda bisa menggunakan kotak pencarian di bagian atas untuk membatasi hasil pada daftar layanan. Anda bisa memilih hanya satu layanan pada blok izin editor visual. Untuk memberikan akses ke lebih dari satu layanan, tambahkan beberapa blok izin dengan memilih Tambahkan lebih banyak izin.
5. Di Tindakan yang diizinkan, pilih tindakan yang akan ditambahkan ke kebijakan. Anda bisa memilih tindakan dengan cara berikut:

   - Pilih kotak centang untuk semua tindakan.

- Pilih tambah tindakan untuk mengetik nama tindakan tertentu. Anda bisa menggunakan wildcards (*) untuk menentukan beberapa tindakan.
- Pilih satu grup Tingkat akses untuk memilih semua tindakan untuk tingkat akses tersebut (misalnya, Baca, Tulis, atau Daftar).
- Perluas setiap grup Tingkat akses untuk memilih tindakan individu.

Secara default, kebijakan yang Anda buat mengizinkan tindakan yang Anda pilih. Sebaliknya, untuk menolak tindakan terpilih, pilih Beralih ke menolak izin. Karena IAMmenolak secara default, kami sarankan sebagai praktik terbaik keamanan bahwa Anda mengizinkan izin hanya untuk tindakan dan sumber daya yang dibutuhkan pengguna. Anda harus membuat JSON pernyataan untuk menolak izin hanya jika Anda ingin mengganti izin secara terpisah yang diizinkan oleh pernyataan atau kebijakan lain. Kami sarankan Anda membatasi jumlah izin penolakan seminim mungkin karena dapat meningkatkan kesulitan izin pemecahan masalah.

6. Untuk Sumber Daya, bila layanan dan tindakan yang Anda pilih di langkah sebelumnya tidak mendukung pilihan sumber daya tertentu, semua sumber daya diperbolehkan dan Anda tidak bisa mengedit bagian ini.

   Jika Anda memilih satu atau lebih tindakan yang mendukungizin tingkat sumber daya, maka editor visual akan mendaftar sumber daya tersebut. Kemudian Anda bisa memperluas Sumber Daya untuk menentukan sumber daya bagi kebijakan Anda.

   Anda dapat menentukan sumber daya dengan cara berikut:

   - Pilih Tambah ARNs untuk menentukan sumber daya berdasarkan Nama Sumber Daya Amazon mereka (ARN). Anda dapat menggunakan ARN editor visual atau daftar ARNs secara manual. Untuk informasi selengkapnya tentang ARN sintaks, lihat Amazon Resource Name (ARN) di Referensi Umum AWS Panduan. Untuk informasi tentang penggunaan ARNs dalam `Resource` elemen kebijakan, lihatIAMJSONelemen kebijakan: Resource.
   - Pilih Apa saja di akun ini di samping sumber daya untuk memberikan izin ke sumber daya apa pun dari jenis itu.
   - Pilih Semua untuk memilih semua sumber daya untuk layanan ini.

7. (Opsional) Pilih Ketentuan permintaan - opsional untuk menambahkan kondisi ke kebijakan yang Anda buat. Ketentuan membatasi efek pernyataan JSON kebijakan. Misalnya, Anda dapat menentukan bahwa pengguna diizinkan melakukan tindakan pada sumber daya hanya ketika permintaan pengguna tersebut terjadi di rentang waktu tertentu. Anda juga dapat menggunakan kondisi yang umum digunakan untuk membatasi apakah pengguna harus diautentikasi

menggunakan perangkat multi-faktor authentication ()MFA. Atau Anda bisa meminta agar permintaan yang berasal dari rentang alamat IP tertentu. Untuk daftar semua kunci konteks yang dapat Anda gunakan dalam kondisi kebijakan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan di Referensi Otorisasi Layanan.

Anda bisa memilih kondisi dengan cara berikut:

- Gunakan kotak centang untuk memilih kondisi yang biasa dipakai.
- Pilih Tambahkan kondisi lain untuk menentukan kondisi lain. Pilih kondisi dari Kunci Kondisi, Pengukur, dan Operator, lalu ketik Nilai. Untuk menambahkan lebih dari satu nilai, pilih Tambah. Anda dapat mempertimbangkan nilai tersebut terhubung secara logika "ATAU" operator. Setelah selesai, pilih Tambahkan kondisi.

Untuk menambahkan lebih dari satu kondisi, pilih Tambahkan kondisi lain lagi. Ulangi seperlunya. Setiap kondisi berlaku hanya untuk blok izin editor visual yang satu ini. Semua kondisi haruslah benar agar blok izin dapat dianggap cocok. Dengan kata lain, pertimbangkan kondisi yang akan dihubungkan oleh operator AND "" logis.

Untuk informasi lebih lanjut mengenai elemen Kondisi, lihat IAMJSONelemen kebijakan: Condition di IAMJSONreferensi kebijakan.

8. Untuk menambahkan lebih banyak blok izin, pilih Tambahkan lebih banyak izin. Untuk setiap blok, ulangi langkah 2 sampai 5.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

9. (Opsional) Saat Anda membuat atau mengedit kebijakan di AWS Management Console, Anda dapat membuat JSON atau templat YAML kebijakan yang dapat Anda gunakan dalam AWS CloudFormation templat.

   Untuk melakukan ini, di editor Kebijakan pilih Tindakan, lalu pilih Buat CloudFormation templat. Untuk mempelajari selengkapnya, AWS CloudFormation lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan AWS CloudFormation Pengguna.

10. Setelah selesai menambahkan izin ke kebijakan, pilih Berikutnya.
11. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau Izin yang ditentukan dalam kebijakan ini untuk memastikan bahwa Anda telah memberikan izin yang dimaksud.
12. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihatTag untuk AWS Identity and Access Management sumber daya.
13. Pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.

Setelah Anda membuat kebijakan, Anda dapat melampirkannya ke grup, pengguna, atau peran Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Mengimpor kebijakan terkelola yang ada

Cara mudah untuk membuat kebijakan baru adalah dengan mengimpor kebijakan terkelola yang ada di dalam akun Anda yang setidaknya memiliki beberapa izin yang Anda perlukan. Anda kemudian bisa mengubah kebijakan tersebut untuk menyesuaikannya dengan persyaratan baru Anda.

Anda tidak bisa mengimpor kebijakan inline. Untuk mempelajari perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Untuk mengimpor kebijakan terkelola yang sudah ada di editor visual

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di editor Kebijakan, pilih Visual dan kemudian di sisi kanan halaman, pilih Tindakan lalu pilih Kebijakan impor.
5. Di jendela Kebijakan impor, pilih kebijakan terkelola yang paling cocok dengan kebijakan yang ingin Anda sertakan dalam kebijakan baru Anda. Anda dapat menggunakan kotak pencarian di bagian atas untuk membatasi hasil dalam daftar kebijakan.
6. Pilih kebijakan Impor.

   Kebijakan yang diimpor ditambahkan pada blok izin baru di bagian bawah kebijakan Anda.
7. Gunakan editor Visual atau pilih JSONuntuk menyesuaikan kebijakan Anda. Lalu pilih Berikutnya.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

8. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Anda tidak bisa mengedit pengaturan kemudian. Tinjau Izin yang ditentukan dalam kebijakan ini, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

Untuk mengimpor kebijakan terkelola yang ada di JSONeditor

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih JSONopsi, lalu di sisi kanan halaman, pilih Tindakan lalu pilih Kebijakan impor.
5. Di jendela Kebijakan impor, pilih kebijakan terkelola yang paling cocok dengan kebijakan yang ingin Anda sertakan dalam kebijakan baru Anda. Anda dapat menggunakan kotak pencarian di bagian atas untuk membatasi hasil dalam daftar kebijakan.
6. Pilih kebijakan Impor.

   Pernyataan dari kebijakan yang diimpor ditambahkan ke bagian bawah JSON kebijakan Anda.
7. Sesuaikan kebijakan Anda diJSON. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya. Sesuaikan kebijakan Anda di JSON, atau pilih Visual editor (Editor visual). Lalu pilih Berikutnya.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

8. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Anda tidak bisa mengedit hal ini kemudian. Tinjau kebijakan Izin yang ditentukan dalam kebijakan ini, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

Setelah Anda membuat kebijakan, Anda dapat melampirkannya ke grup, pengguna, atau peran Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Buat IAM kebijakan (AWS CLI)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS CLI untuk membuat kebijakan yang dikelola pelanggan diIAM. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiri. Akun AWS Sebagai praktik terbaik, kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi IAM kebijakan Anda guna memastikan izin yang aman dan fungsional. Dengan memvalidasi kebijakan Anda, Anda dapat mengatasi kesalahan atau rekomendasi apa pun sebelum Anda melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda. Akun AWS

Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Membuat IAM kebijakan (AWS CLI)

Anda dapat membuat kebijakan terkelola IAM pelanggan atau kebijakan inline menggunakan AWS Command Line Interface (AWS CLI).

Untuk membuat kebijakan terkelola pelanggan (AWS CLI)

Gunakan perintah berikut ini:

- buat-kebijakan

Untuk membuat kebijakan inline untuk IAM identitas (grup, pengguna, atau peran) (AWS CLI)

Gunakan salah satu perintah berikut:

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> Anda tidak dapat menggunakan IAM untuk menyematkan kebijakan sebaris untuk peran terkait layanan.

Untuk memvalidasi kebijakan terkelola pelanggan (AWS CLI)

Gunakan perintah IAM Access Analyzer berikut:

- validasi-kebijakan

Buat IAM kebijakan (AWS API)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS API untuk membuat kebijakan yang dikelola pelanggan diIAM. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiri. Akun AWS Sebagai praktik terbaik, kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi IAM kebijakan Anda guna memastikan izin yang aman dan fungsional. Dengan memvalidasi kebijakan Anda, Anda dapat mengatasi kesalahan atau rekomendasi apa pun sebelum Anda melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda. Akun AWS

Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Membuat IAM kebijakan (AWS API)

Anda dapat membuat kebijakan terkelola IAM pelanggan atau kebijakan inline menggunakan. AWS API

Untuk membuat kebijakan terkelola pelanggan (AWS API)

Hubungi operasi berikut ini:

- CreatePolicy

Untuk membuat kebijakan inline untuk IAM identitas (grup, pengguna, atau peran) (AWS API)

Panggil salah satu operasi berikut ini:

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> Anda tidak dapat menggunakan IAM untuk menyematkan kebijakan sebaris untuk peran terkait layanan.

Untuk memvalidasi kebijakan yang dikelola pelanggan ()AWS API

Hubungi operasi IAM Access Analyzer berikut:

- ValidatePolicy

Validasi kebijakan IAM

Kebijakan adalah JSON dokumen yang menggunakan tata bahasa IAM kebijakan. Saat Anda melampirkan kebijakan ke IAM entitas, seperti pengguna, grup, atau peran, kebijakan tersebut akan memberikan izin kepada entitas tersebut.

Saat Anda membuat atau mengedit kebijakan kontrol IAM akses menggunakan AWS Management Console, secara AWS otomatis memeriksanya untuk memastikan bahwa IAM kebijakan tersebut mematuhi tata bahasa kebijakan. Jika AWS menentukan sebuah kebijakan tidak sesuai dengan tata bahasa, ia akan meminta Anda memperbaiki kebijakan tersebut.

IAMAccess Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat Validasi kebijakan IAMAccess Analyzer. Untuk melihat daftar peringatan, kesalahan, dan saran yang ditampilkan oleh IAM Access Analyzer, lihat referensi pemeriksaan kebijakan IAM Access Analyzer.

Cakupan validasi

AWS memeriksa sintaks JSON kebijakan dan tata bahasa. Ini juga memverifikasi bahwa Anda ARNs diformat dengan benar dan nama tindakan dan kunci kondisi sudah benar.

Mengakses validasi kebijakan

Kebijakan divalidasi secara otomatis saat Anda membuat JSON kebijakan atau mengedit kebijakan yang ada di. AWS Management Console Jika sintaks kebijakan tidak valid, Anda menerima notifikasi dan harus memperbaiki masalahnya sebelum bisa melanjutkan. Temuan dari validasi kebijakan IAM Access Analyzer secara otomatis ditampilkan di AWS Management Console jika Anda memiliki izin untuk. `access-analyzer:ValidatePolicy` Anda juga dapat memvalidasi kebijakan menggunakan AWS API atau AWS CLI.

Kebijakan yang sudah Ada

Anda mungkin memiliki kebijakan yang sudah ada yang tidak valid karena dibuat atau terakhir disimpan sebelum pembaruan terbaru ke mesin kebijakan. Sebagai praktik terbaik, kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi IAM kebijakan Anda guna memastikan izin yang aman dan fungsional. Kami menyarankan Anda membuka kebijakan yang ada dan meninjau hasil validasi kebijakan yang dihasilkan. Anda tidak dapat mengedit dan menyimpan kebijakan yang sudah ada tanpa memperbaiki kesalahan sintaks kebijakan apa pun.

IAMpengujian kebijakan dengan simulator IAM kebijakan

Untuk informasi selengkapnya tentang cara dan mengapa menggunakan IAM kebijakan, lihatKebijakan dan izin di AWS Identity and Access Management.

Anda dapat mengakses Konsol Simulator IAM Kebijakan di: https://policysim.aws.amazon.com/

> **Important**
>
> Hasil simulator kebijakan dapat berbeda dari AWS lingkungan hidup Anda. Kami menyarankan Anda memeriksa kebijakan Anda terhadap AWS lingkungan hidup Anda setelah pengujian menggunakan simulator kebijakan untuk mengonfirmasi bahwa Anda memiliki hasil yang diinginkan. Untuk informasi selengkapnya, lihat Cara kerja simulator kebijakan IAM.

Memulai dengan Simulator IAM Kebijakan

Dengan simulator IAM kebijakan, Anda dapat menguji dan memecahkan masalah batasan kebijakan dan izin berbasis identitas. IAM Berikut adalah beberapa hal umum yang bisa Anda lakukan dengan simulator kebijakan:

- Uji kebijakan berbasis identitas yang dilampirkan ke IAM pengguna, IAM grup, atau peran dalam Anda. Akun AWS Jika lebih dari satu kebijakan terlampir ke pengguna, grup pengguna, atau peran, Anda bisa menguji semua kebijakan, atau memilih kebijakan tertentu untuk diuji. Anda bisa menguji tindakan mana yang diizinkan atau ditolak oleh kebijakan terpilih untuk sumber daya tertentu.
- Uji dan pecahkan masalah efek batas izin pada entitas. IAM Anda hanya dapat mensimulasikan satu batas izin pada satu waktu.
- Uji efek kebijakan berbasis sumber daya pada IAM pengguna yang dilampirkan ke AWS sumber daya, seperti bucket Amazon S3, antrian Amazon, topik Amazon, atau kubah Amazon S3 SQS Glacier. SNS Untuk menggunakan kebijakan berbasis sumber daya dalam simulator kebijakan bagi IAM pengguna, Anda harus menyertakan sumber daya dalam simulasi. Anda juga harus memilih kotak centang untuk menyertakan kebijakan sumber daya tersebut dalam simulasi.

  > **Note**
  >
  > Simulasi kebijakan berbasis sumber daya tidak didukung untuk peran. IAM

- Jika Anda Akun AWS adalah anggota organisasi di AWS Organizations, maka Anda dapat menguji dampak kebijakan kontrol layanan (SCPs) pada kebijakan berbasis identitas Anda.

  > **Note**
  >
  > Simulator kebijakan tidak mengevaluasi SCPs yang memiliki kondisi apa pun.

- Uji kebijakan berbasis identitas baru yang belum dilampirkan ke pengguna, grup pengguna, atau peran dengan mengetik atau menyalinnya ke dalam simulator kebijakan. Semua ini hanya digunakan dalam simulasi dan tidak disimpan. Anda tidak dapat mengetik atau menyalin kebijakan berbasis sumber daya di simulator kebijakan.
- Uji kebijakan berbasis identitas dengan layanan, tindakan, dan sumber daya yang dipilih. Misalnya, Anda bisa menguji untuk memastikan kebijakan Anda mengizinkan entitas untuk melakukan `ListAllMyBuckets`, `CreateBucket`, dan `DeleteBucket` tindakan di layanan Amazon S3 di bucket tertentu.
- Simulasikan skenario dunia nyata dengan menyediakan kunci konteks, misalnya alamat IP atau tanggal, yang disertakan ke`Condition` elemn dalam kebijakan yang diuji.

> **Note**
>
> Simulator kebijakan tidak mensimulasikan tag yang disediakan sebagai input jika kebijakan berbasis identitas dalam simulasi tidak memiliki `Condition` elemen yang secara eksplisit memeriksa tag.

- Identifikasi pernyataan spesifik mana dalam kebijakan berbasis identitas yang menghasilkan mengizinkan atau menolak akses ke sumber daya atau tindakan tertentu.

Topik

Cara kerja simulator kebijakan IAM

Simulator kebijakan mengevaluasi pernyataan dalam kebijakan berbasis identitas dan masukan yang Anda berikan selama simulasi. Hasil simulator kebijakan dapat berbeda dari AWS lingkungan hidup Anda. Kami menyarankan Anda memeriksa kebijakan Anda terhadap AWS lingkungan hidup Anda setelah pengujian menggunakan simulator kebijakan untuk mengonfirmasi bahwa Anda memiliki hasil yang diinginkan.

Simulator kebijakan berbeda dari AWS lingkungan hidup dengan cara berikut:

- Simulator kebijakan tidak membuat permintaan AWS layanan yang sebenarnya, sehingga Anda dapat menguji permintaan dengan aman yang mungkin membuat perubahan yang tidak diinginkan pada AWS lingkungan hidup Anda. Simulator kebijakan tidak mempertimbangkan nilai kunci konteks nyata dalam produksi.
- Karena simulator kebijakan tidak mensimulasikan menjalankan tindakan yang dipilih, simulator kebijakan tidak dapat melaporkan respons apa pun terhadap permintaan yang disimulasikan. Satu-satunya hasil yang dikembalikan adalah apakah tindakan yang diminta akan diizinkan atau ditolak.
- Jika Anda mengedit kebijakan di simulator kebijakan, perubahan ini hanya memengaruhi simulator kebijakan. Kebijakan terkait di Anda Akun AWS tetap tidak berubah.
- Anda tidak dapat menguji kebijakan kontrol layanan (SCPs) dengan kondisi apa pun.

- Simulator kebijakan tidak mendukung simulasi IAM peran dan pengguna untuk akses lintas akun.

> **Note**
>
> Simulator IAM kebijakan tidak menentukan layanan mana yang mendukung kunci kondisi global untuk otorisasi. Misalnya, simulator kebijakan tidak mengidentifikasi bahwa layanan tidak mendukung `aws:TagKeys`.

Izin yang diperlukan untuk menggunakan simulator IAM kebijakan

Anda dapat menggunakan konsol simulator kebijakan atau simulator kebijakan API untuk menguji kebijakan. Secara default, pengguna konsol dapat menguji kebijakan yang belum dilampirkan ke pengguna, grup pengguna, atau peran dengan mengetik atau menyalin kebijakan tersebut ke dalam simulator kebijakan. Kebijakan ini hanya dipakai dalam simulasi dan tidak mengungkap informasi sensitif. APIpengguna harus memiliki izin untuk menguji kebijakan yang tidak dilampirkan. Anda dapat mengizinkan konsol atau API pengguna untuk menguji kebijakan yang dilampirkan ke IAM pengguna, IAM grup, atau peran dalam Anda Akun AWS. Untuk melakukannya, Anda harus mengizinkan pengambilan kebijakan tersebut. Guna menguji kebijakan berbasis sumber daya, pengguna harus berizin untuk mengambil kebijakan sumber daya tersebut.

Untuk contoh konsol dan API kebijakan yang memungkinkan pengguna mensimulasikan kebijakan, lihatthe section called “Contoh kebijakan: AWS Identity and Access Management (IAM)”.

Izin diperlukan untuk menggunakan konsol simulator kebijakan

Anda dapat mengizinkan pengguna untuk menguji kebijakan yang dilampirkan ke IAM pengguna, IAM grup, atau peran dalam Anda Akun AWS. Untuk melakukannya, Anda harus mengizinkan pengguna Anda untuk mengambil kebijakan tersebut. Guna menguji kebijakan berbasis-sumber daya, pengguna harus berizin untuk mengambil kebijakan sumber daya tersebut.

Untuk melihat kebijakan contoh yang mengizinkan penggunaan konsol simulator kebijakan untuk kebijakan yang melekat ke pengguna, grup pengguna, atau peran, lihat IAM: Akses konsol simulator kebijakan.

Untuk melihat kebijakan contoh yang memungkinkan penggunaan konsol simulator kebijakan hanya untuk pengguna dengan jalur tertentu, lihat IAM: Akses konsol simulator kebijakan berdasarkan jalur pengguna.

Untuk membuat kebijakan yang memungkinkan penggunaan konsol simulator kebijakan hanya untuk satu tipe entitas, gunakan prosedur berikut.

Untuk memungkinkan pengguna konsol mensimulasikan kebijakan bagi pengguna

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetUser`
- `iam:GetUserPolicy`
- `iam:ListAttachedUserPolicies`
- `iam:ListGroupsForUser`
- `iam:ListGroupPolicies`
- `iam:ListUserPolicies`
- `iam:ListUsers`

Untuk memungkinkan pengguna konsol mensimulasikan kebijakan untuk grup IAM

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetGroup`
- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:ListAttachedGroupPolicies`
- `iam:ListGroupPolicies`
- `iam:ListGroups`

Untuk memungkinkan pengguna konsol mensimulasikan kebijakan untuk peran

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetPolicy`

- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:GetRolePolicy`
- `iam:ListAttachedRolePolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`

Untuk menguji kebijakan berbasis-sumber daya, pengguna harus berizin untuk mengambil kebijakan sumber daya tersebut.

Untuk memungkinkan pengguna konsol menguji kebijakan berbasis sumber daya di bucket Amazon S3

Sertakan tindakan berikut dalam kebijakan Anda:

- `s3:GetBucketPolicy`

Misalnya, kebijakan berikut memakai tindakan ini untuk memungkinkan pengguna konsol mensimulasikan kebijakan berbasis sumber daya dalam bucket Amazon S3 tertentu.

```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "s3:GetBucketPolicy",
            "Resource":"arn:aws:s3:::bucket-name/*"
          }
        ]
      }
```

Izin yang diperlukan untuk menggunakan simulator kebijakan API

Simulator kebijakan API beroperasi GetContextKeyForCustomPolicydan SimulateCustomPolicymemungkinkan Anda untuk menguji kebijakan yang belum dilampirkan ke pengguna, grup pengguna, atau peran. Untuk menguji kebijakan tersebut, Anda meneruskan kebijakan sebagai string ke. API Kebijakan ini hanya dipakai dalam simulasi dan tidak mengungkap informasi sensitif. Anda juga dapat menggunakan kebijakan API untuk menguji yang dilampirkan

ke IAM pengguna, IAM grup, atau peran di situs Anda Akun AWS. Untuk melakukan itu, Anda harus memberi pengguna izin untuk menelepon GetContextKeyForPrincipalPolicydan SimulatePrincipalPolicy.

Untuk melihat contoh kebijakan yang memungkinkan penggunaan simulator kebijakan API untuk kebijakan terlampir dan tidak terikat saat ini Akun AWS, lihatIAM: Akses API simulator kebijakan.

Untuk membuat kebijakan yang memungkinkan penggunaan simulator kebijakan hanya API untuk satu jenis kebijakan, gunakan prosedur berikut.

Untuk memungkinkan API pengguna mensimulasikan kebijakan yang diteruskan langsung ke string API as

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetContextKeysForCustomPolicy`
- `iam:SimulateCustomPolicy`

Untuk memungkinkan API pengguna mensimulasikan kebijakan yang dilampirkan pada IAM pengguna, IAM grup, peran, atau sumber daya

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetContextKeysForPrincipalPolicy`
- `iam:SimulatePrincipalPolicy`

Misalnya, untuk memberi pengguna bernama Bob izin menyimulasikan kebijakan yang ditetapkan untuk pengguna bernama Alice, beri Bob akses ke sumber daya berikut ini: `arn:aws:iam::777788889999:user/alice`.

Untuk melihat contoh kebijakan yang memungkinkan penggunaan simulator kebijakan API hanya untuk pengguna dengan jalur tertentu, lihatIAM: Akses API simulator kebijakan berdasarkan jalur pengguna.

Menggunakan simulator kebijakan IAM (konsol)

Secara default, pengguna konsol dapat menguji kebijakan yang belum dilampirkan ke pengguna, grup pengguna, atau peran dengan mengetik atau menyalin kebijakan tersebut ke konsol simulator kebijakan. Kebijakan ini hanya dipakai dalam simulasi dan tidak mengungkap informasi sensitif.

Untuk menguji kebijakan yang tidak melekat ke pengguna, grup pengguna, atau peran (konsole)

1. Buka konsol simulator IAM kebijakan di: https://policysim.aws.amazon.com/.
2. Di menu Mode: pada bagian atas halaman, pilih Kebijakan Baru.
3. Di Policy Sandbox, pilih Buat Kebijakan Baru.
4. Ketik atau salin kebijakan ke dalam simulator kebijakan, dan gunakan simulator kebijakan seperti yang dijelaskan dalam langkah-langkah berikut.

Setelah memiliki izin untuk menggunakan Konsol Simulator IAM Kebijakan, Anda dapat menggunakan simulator kebijakan untuk menguji kebijakan IAM pengguna, grup pengguna, peran, atau sumber daya.

Untuk menguji kebijakan yang melekat ke pengguna, grup pengguna, atau peran (konsole)

1. Buka konsol simulator IAM kebijakan di https://policysim.aws.amazon.com/.

   > **Note**
   >
   > Untuk masuk ke simulator kebijakan sebagai IAM pengguna, gunakan login unik Anda URL untuk masuk ke AWS Management Console. Lalu pergi ke https://policysim.aws.amazon.com/. Untuk informasi selengkapnya tentang masuk sebagai IAM pengguna, lihatCara IAM pengguna masuk AWS.

   Simulator kebijakan terbuka dalam mode Kebijakan yang Ada dan mencantumkan IAM pengguna di akun Anda di bawah Pengguna, Grup, dan Peran.
2. Pilih opsi yang sesuai dengan tugas Anda:

Untuk menguji ini:	Lakukan ini:
Kebijakan yang melekat pada pengguna	Pilih Pengguna pada daftarPengguna, Grup, dan Peran. Lalu pilih pengguna.
Kebijakan yang melekat pada grup pengguna	Pilih Grup pada daftarPengguna, Grup, dan Peran. Lalu pilih grup pengguna.

Untuk menguji ini:	Lakukan ini:
Kebijakan yang melekat pada peran	Pilih Peran pada daftarPengguna, Grup, dan Peran. Lalu pilih peran.
Kebijakan yang terlampir pada sumber daya	Lihat Step 9.
Kebijakan khusus untuk pengguna, grup pengguna, atau peran	Pilih Buat Kebijakan Baru. Pada panel Kebijakan baru, ketik atau tempelkan kebijakan dan pilih Terapkan.

> **Kiat**
>
> Untuk menguji kebijakan yang dilampirkan ke grup pengguna, Anda dapat meluncurkan simulator IAM kebijakan langsung dari IAMkonsol: Di panel navigasi, pilih Grup pengguna. Pilih nama grup yang dimana Anda ingin menguji kebijakan, dan kemudian pilih tabIzin. Pilih Simulasi.
> Untuk menguji kebijakan yang dikelola pelanggan yang terlampir pada pengguna: Di panel navigasi, pilih Pengguna. Pilih nama pengguna dengan siapa Anda ingin menguji kebijakan. Lalu pilih tabIzin dan perluas kebijakan yang ingin Anda uji. Di ujung kanan, pilih Simulasikan kebijakan. Simulator IAM Kebijakan terbuka di jendela baru dan menampilkan kebijakan yang dipilih di panel Kebijakan.

3. (Opsional) Jika akun Anda adalah anggota organisasi AWS Organizations, pilih kotak centang di samping AWS Organizations SCPsuntuk menyertakan SCPs dalam evaluasi simulasi Anda. SCPsadalah JSON kebijakan yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU). SCPMembatasi izin untuk entitas di akun anggota. Jika SCP memblokir layanan atau tindakan, maka tidak ada entitas di akun itu yang dapat mengakses layanan tersebut atau melakukan tindakan itu. Hal ini berlaku bahkan jika administrator secara eksplisit memberikan izin untuk layanan atau tindakan tersebut melalui kebijakan atau sumber daya. IAM

   Jika akun Anda bukan anggota sebuah organisasi, maka kotak centang tidak akan muncul.

4. (Opsional) Anda dapat menguji kebijakan yang ditetapkan sebagai batas izin untuk IAM entitas (pengguna atau peran), tetapi tidak untuk grup. IAM Jika kebijakan batasan izin saat ini diatur

untuk entitas, ia muncul dalam panel Kebijakan. Anda hanya bisa mengatur satu batasan izin untuk sebuah entitas. Untuk menguji batasan izin yang berbeda, Anda bisa membuat batasan izin khusus. Untuk melakukannya, pilih Buat Kebijakan Baru. Panel Kebijakan baru akan terbuka. Di menu, pilih Kebijakan Batas IAM Izin Kustom. Masukkan nama untuk kebijakan baru dan ketik atau salin kebijakan ke ruang di bawah ini. Pilih Terapkan untuk menyimpan kebijakan tersebut. Selanjutnya, pilih Kembali untuk kembali ke panel Kebijakan awal. Lalu pilih kotak centang di samping batasan izin yang ingin Anda pakai untuk simulasi.

5. (Opsional) Anda hanya bisa menguji subset kebijakan yang melekat pada pengguna, grup pengguna, atau peran. Untuk melakukannya, di panelKebijakan kosongkan kotak centang di samping setiap kebijakan ingin Anda keluarkan.

6. Di bawah Simulator Kebijakan, pilih Pilih layanan lalu pilih layanan yang ingin diuji. Lalu pilih Pilih tindakan dan pilih satu atau lebih tindakan untuk diuji. Walaupun menu menunjukkan pilihan yang tersedia hanya untuk satu layanan pada satu waktu, semua layanan dan tindakan yang telah Anda pilih muncul di Pengaturan Tindakan dan Hasil.

7. (Opsional) Jika salah satu kebijakan yang Anda pilih Step 2 dan Step 5 menyertakan kondisi dengan kunci kondisi AWS global, berikan nilai untuk kunci tersebut. Anda melakukan hal ini dengan memperluas bagianPengaturan Global dan mengetikkan nilai untuk nama utama yang ditampilkan di sana.

   > **Warning**
   >
   > Jika Anda membiarkan nilai untuk kunci kondisi kosong, maka kunci tersebut akan diabaikan selama simulasi. Pada beberapa kasus, hal ini menghasilkan kesalahan, dan simulasi gagal untuk dijalankan. Pada kasus lain, simulasi berjalan, namun hasilnya mungkin tidak bisa diandalkan. Pada kasus tersebut, simulasi tidak sesuai dengan kondisi dunia nyata yang mencakup nilai untuk kunci kondisi atau variabel.

8. (Opsional) Setiap tindakan terpilih muncul di daftarPengaturan Tindakan dan Hasil denganTidak disimulasikan muncul di kolomIzin sampai Anda benar-benar menjalankan simulasi tersebut. Sebelum menjalankan simulasi, Anda bisa mengonfigurasi setiap tindakan dengan sumber daya. Untuk mengonfigurasi tindakan individu untuk skenario tertentu, pilih panah untuk memperluas baris tindakan. Jika tindakan mendukung izin tingkat sumber daya, Anda dapat mengetikkan Amazon Resource Name (ARN) sumber daya tertentu yang aksesnya ingin Anda uji. Secara default, setiap sumber daya diatur sebagai wildcard (*). Anda juga bisa menentukan nilai tertentu untuk kunci konteks kondisi mana pun. Seperti disebut sebelumnya, kunci dengan nilai kosong diabaikan, yang bisa menimbulkan kegagalan simulasi atau hasil yang tidak bisa diandalkan.

a. Pilih panah di samping nama tindakan untuk memperluas setiap baris dan mengonfigurasi tambahan informasi yang diperlukan untuk secara akurat menyimulasikan tindakan pada skenario Anda. Jika tindakan memerlukan izin tingkat sumber daya apa pun, Anda dapat mengetikkan Amazon Resource Name (ARN) sumber daya tertentu yang ingin Anda simulasikan aksesnya. Secara default, setiap sumber daya diatur sebagai wildcard (*).

b. Bila tindakan tersebut mendukung izin tingkat sumber daya namun tidak memerlukannya, maka Anda bisa memilih Tambah Sumber Daya untuk memilih tipe sumber daya yang ingin Anda tambahkan ke simulasi.

c. Jika salah satu kebijakan terpilih mencakup `Condition` elemen yang merujuk ke kunci konteks untuk layanan tindakan ini, maka nama kunci tersebut akan muncul di bawah tindakan. Anda dapat menetapkan nilai yang akan dipakai selama simulasi tindakan tersebut untuk sumber daya yang ditentukan.

Tindakan yang memerlukan grup tipe sumber daya berbeda

Beberapa tindakan memerlukan tipe sumber daya yang berbeda dengan keadaan berbeda. Setiap grup tipe sumber daya terkait dengan sebuah skenario. Jika salah satu dari ini berlaku untuk simulasi Anda, pilih dan simulator kebijakan memerlukan jenis sumber daya yang sesuai untuk skenario itu. Daftar berikut menunjukkan setiap opsi skenario yang didukung dan sumber daya yang harus Anda tentukan untuk menjalankan simulasi tersebut.

Setiap EC2 skenario Amazon berikut mengharuskan Anda menentukan`instance`,`image`, dan `security-group` sumber daya. Jika skenario Anda menyertakan EBS volume, maka Anda harus menentukannya `volume` sebagai sumber daya. Jika EC2 skenario Amazon menyertakan virtual private cloud (VPC), maka Anda harus menyediakan `network-interface` sumber daya. Jika termasuk IP subnet, maka Anda harus menentukan `subnet` sumber dayanya. Untuk informasi selengkapnya tentang opsi EC2 skenario Amazon, lihat Platform yang Didukung di Panduan EC2 Pengguna Amazon.

- EC2-VPC-InstanceStore

  instance, image, security-group, network-interface

- EC2- VPC - InstanceStore -Subnet

  instance, image, security-group, network-interface, subnet

- EC2-VPC-EBS

instance, image, security-group, network-interface, volume

- EC2- VPC - EBS -Subnet

  instance, image, security-group, network-interface, subnet, volume

9. (Opsional) Jika Anda ingin memasukkan kebijakan berbasis sumber daya dalam simulasi Anda, maka terlebih dahulu Anda harus memilih tindakan yang ingin Anda simulasikan pada sumber daya tersebut dalam Step 6. Perluas baris untuk tindakan yang dipilih, dan ARN ketik sumber daya dengan kebijakan yang ingin Anda simulasikan. Kemudian pilih Sertakan Kebijakan Sumber Daya di sebelah kotak ARNteks. Simulator IAM kebijakan saat ini mendukung kebijakan berbasis sumber daya hanya dari layanan berikut: Amazon S3 (hanya kebijakan berbasis sumber daya; saat ini tidak didukung), Amazon, Amazon, dan brankas S3 Glacier yang tidak terkunci (SQSbrankas terkunci ACLs saat ini tidak didukung). SNS
10. Pilih Jalankan Simulasi di sudut kanan atas.

    Izin kolom di setiap baris Pengaturan Tindakan dan Hasil menampilkan hasil simulasi dari tindakan tersebut di sumber daya yang ditentukan.
11. Untuk melihat pernyataan mana dalam sebuah kebijakan yang mengizinkan atau menolak tindakan, pilih ***N*** pernyataan (-pernyataan) yang cocok tautkan di kolom Izin untuk memperluas baris. Lalu pilih tautanTampilkan pernyataan. Panel Kebijakan menunjukkan kebijakan yang relevan dengan pernyataan yang memengaruhi hasil simulasi yang disoroti.

> **Note**
>
> Jika sebuah tindakan secara implisit ditolak—yaitu, jika tindakan tersebut ditolak hanya karena tidak secara eskplisit diizinkan—opsi Daftar dan Tampilkan pernyataan tidak ditampilkan.

Pemecahan masalah pesan konsol simulator kebijakan IAM

Tabel berikut mencantumkan pesan informasi dan peringatan yang mungkin Anda temui saat menggunakan simulator IAM kebijakan. Tabel ini juga memberikan langkah-langkah yang bisa Anda ambil untuk menyelesaikannya.

Pesan	Langkah-langkah untuk menyelesaikan
Kebijakan ini telah diedit Perubahan tidak akan disimpan ke akun Anda.	Tidak ada tindakan yang diperlukan. Pesan ini bersifat informatif Jika Anda mengedit kebijakan yang ada di simulator IAM kebijakan , perubahan Anda tidak akan memengaru hi kebijakan Anda Akun AWS. Simulator kebijakan memungkinkan Anda membuat perubahan pada kebijakan hanya untuk tujuan pengujian.
Tidak bisa mengambil kebijakan sumber daya. Alasan: *detailed error message*	Simulator kebijakan tidak dapat mengakses kebijakan berbasis sumber daya yang diminta. Pastikan sumber daya yang ditentukan ARN sudah benar dan pengguna yang menjalankan simulasi memiliki izin untuk membaca kebijakan sumber daya.
Satu atau lebih kebijakan memerlukan nilai dalam pengaturan simulasi. Simulasi bisa gagal tanpa nilai ini.	Pesan ini muncul jika kebijakan yang sedang Anda uji berisi kunci kondisi atau variabel namun Anda tidak memberi nilai apa pun untuk kunci atau variabel tersebut diPengaturan Simulasi. Untuk mengabaikan pesan ini, pilih Pengatura n Simulasi, Lalu masukkan nilai untuk setiap kunci kondisi atau variabel.
Anda telah mengubah kebijakan. Hasil ini tidak lagi valid.	Pesan ini muncul jika Anda telah mengubah kebijakan yang dipilih ketika hasil ditampilkan di panelHasil. Hasil yang ditampilkan di panel Hasil tidak diperbarui secara dinamis. Untuk mengabaikan pesan ini, pilih Jalankan Simulasi lagi untuk menampilkan hasil simulasi baru berdasarkan perubahan yang dibuat di panelKebijakan.

Pesan	Langkah-langkah untuk menyelesaikan
Sumber daya yang Anda ketikkan untuk simulasi ini tidak cocok dengan layanan ini.	Pesan ini muncul jika Anda telah mengetik Amazon Resource Name (ARN) di panel Pengaturan Simulasi yang tidak cocok dengan layanan yang Anda pilih untuk simulasi saat ini. Misalnya, pesan ini muncul jika Anda menentukan sumber daya Amazon DynamoDB ARN untuk Amazon tetapi Anda memilih Amazon Redshift sebagai layanan yang akan disimulasikan. Untuk mengabaikan pesan ini, lakukan salah satu hal berikut: • Hapus ARN dari kotak di panel Pengaturan Simulasi. • Pilih layanan yang cocok dengan ARN yang Anda tentukan di Pengaturan Simulasi.
Tindakan ini termasuk dalam layanan yang mendukung mekanisme kontrol akses khusus selain kebijakan berbasis sumber daya, seperti kebijakan kunci Amazon S3 ACLs atau S3 Glacier vault. Simulator kebijakan tidak mendukung mekanisme ini, sehingga hasilnya bisa berbeda dari lingkungan produksi Anda.	Tidak ada tindakan yang diperlukan. Pesan ini bersifat informatif Dalam versi saat ini, simulator kebijakan mengevaluasi kebijakan yang dilampirkan pada pengguna dan IAM grup, dan dapat mengevaluasi kebijakan berbasis sumber daya untuk Amazon S3, Amazon, SQS Amazon, SNS dan S3 Glacier. Simulator kebijakan tidak mendukung semua mekanisme kendali akses yang didukung oleh layanan AWS lainnya.

Pesan	Langkah-langkah untuk menyelesaikan
DynamoDB saat ini FGAC tidak didukung.	Tidak ada tindakan yang diperlukan. Pesan informatif ini merujuk ke kendali akses fine-grained. Kontrol akses berbutir halus adalah kemampuan untuk menggunakan kondisi IAM kebijakan untuk menentukan siapa yang dapat mengakses item dan atribut data individual dalam tabel dan indeks DynamoDB. Juga merujuk ke tindakan yang bisa dilakukan pada tabel dan indeks ini. Versi simulator IAM kebijakan saat ini tidak mendukung jenis kondisi kebijakan ini. Untuk informasi selengkapnya tentang kontrol akses berbutir halus DynamoDB, lihat Kontrol Akses Berbutir Baik untuk DynamoDB.
Anda memiliki kebijakan yang tidak mematuhi sintaksis kebijakan. Anda dapat menggunakan validasi kebijakan untuk meninjau pembaruan yang disarankan bagi kebijakan Anda.	Pesan ini muncul di bagian atas daftar kebijakan jika Anda memiliki kebijakan yang tidak sesuai dengan tata bahasa IAM kebijakan . Untuk menyimulasikan kebijakan ini, tinjau opsi validasi kebijakan di Validasi kebijakan IAM untuk mengidentifikasi dan memperbaiki kebijakan ini.
Kebijakan ini harus diperbarui agar mematuhi aturan sintaksis kebijakan terbaru.	Pesan ini ditampilkan jika Anda memiliki kebijakan yang tidak sesuai dengan tata bahasa IAM kebijakan. Untuk menyimulasikan kebijakan ini, tinjau opsi validasi kebijakan di Validasi kebijakan IAM untuk mengidentifikasi dan memperbaiki kebijakan ini.

Menggunakan simulator IAM kebijakan (AWS CLI dan AWS API)

Perintah simulator kebijakan biasanya memerlukan API operasi panggilan untuk melakukan dua hal:

1. Mengevaluasi kebijakan dan mengembalikan daftar kunci konteks yang mereka referensikan. Anda perlu tahu kunci konteks mana yang direferensikan sehingga Anda bisa memasok nilai bagi mereka di langkah berikutnya.
2. Simulasikan kebijakan, berikan daftar tindakan, sumber daya, dan kunci konteks yang digunakan selama simulasi.

Untuk alasan keamanan, API operasi telah dipecah menjadi dua kelompok:

- APIoperasi yang mensimulasikan hanya kebijakan yang diteruskan langsung ke string API as. Set ini termasuk GetContextKeysForCustomPolicy dan SimulateCustomPolicy.
- APIoperasi yang mensimulasikan kebijakan yang dilampirkan ke pengguna, grup IAM pengguna, peran, atau sumber daya tertentu. Karena API operasi ini dapat mengungkapkan rincian izin yang ditetapkan ke IAM entitas lain, Anda harus mempertimbangkan untuk membatasi akses ke operasi iniAPI. Set ini termasuk GetContextKeysForPrincipalPolicy dan SimulatePrincipalPolicy. Untuk informasi selengkapnya tentang membatasi akses ke API operasi, lihatContoh kebijakan: AWS Identity and Access Management (IAM).

Dalam kedua kasus, API operasi mensimulasikan efek dari satu atau lebih kebijakan pada daftar tindakan dan sumber daya. Setiap tindakan dipasangkan dengan setiap sumber daya dan simulasi menentukan apakah kebijakan itu mengizinkan atau menolak tindakan untuk sumber daya tersebut. Anda juga bisa memberi nilai bagi setiap kunci konteks yang menjadi referensi kebijakan Anda. Anda bisa mendapatkan daftar kunci konteks yang referensi kebijakan dengan terlebih dahulu memanggil `GetContextKeysForCustomPolicy` atau `GetContextKeysForPrincipalPolicy`. Jika Anda tidak memberikan nilai untuk kunci konteks, simulasi masih berjalan. Tetapi hasilnya mungkin tidak dapat diandalkan karena simulator kebijakan tidak dapat memasukkan kunci konteks itu dalam evaluasi.

Untuk mendapatkan daftar kunci konteks (AWS CLI, AWS API)

Gunakan hal berikut untuk mengevaluasi daftar kebijakan dan mengembalikan daftar kunci konteks yang dipakai dalam kebijakan.

- AWS CLI: `aws iam get-context-keys-for-custom-policy` dan `aws iam get-context-keys-for-principal-policy`
- AWS API: `GetContextKeysForCustomPolicy`dan `GetContextKeysForPrincipalPolicy`

Untuk mensimulasikan IAM kebijakan (AWS CLI, AWS API)

Gunakan yang berikut ini untuk mensimulasikan IAM kebijakan guna menentukan izin efektif pengguna.

- AWS CLI: `aws iam simulate-custom-policy` dan `aws iam simulate-principal-policy`
- AWS API: `SimulateCustomPolicy`dan `SimulatePrincipalPolicy`

Menambahkan dan menghapus izin identitas IAM

Anda menggunakan kebijakan untuk menentukan izin bagi identitas (pengguna, grup pengguna, atau peran). Anda dapat menambah dan menghapus izin dengan melampirkan dan melepaskan IAM kebijakan untuk identitas menggunakan, AWS Command Line Interface (AWS CLI) AWS Management Console, atau. AWS API Anda juga dapat menggunakan kebijakan untuk menetapkan batas izin hanya untuk entitas (pengguna atau peran) yang menggunakan metode yang sama. Batas izin adalah AWS fitur lanjutan yang mengontrol izin maksimum yang dapat dimiliki entitas.

Topik

- Terminologi
- Lihat aktivitas identitas
- Menambahkan izin identitas IAM (konsol)
- Menghapus izin identitas IAM (konsol)
- Menambahkan Kebijakan IAM (AWS CLI)
- Menghapus kebijakan IAM (AWS CLI)
- Menambahkan IAM kebijakan (AWS API)
- Menghapus IAM kebijakan (AWS API)

Terminologi

Saat Anda mengaitkan kebijakan izin dengan identitas (IAMpengguna, IAM grup, dan IAM peran), terminologi dan prosedur bervariasi tergantung pada apakah Anda bekerja dengan kebijakan terkelola atau sebaris:

- Pasang – Dipakai dengan kebijakan terkelola. Anda melampirkan kebijakan terkelola ke identitas (pengguna, grup pengguna, atau peran). Melampirkan kebijakan menerapkan izin dari kebijakan ke identitas.
- Lepas – Dipakai dengan kebijakan terkelola. Anda melepaskan kebijakan terkelola dari IAM identitas (pengguna, grup pengguna, atau peran). Melepas kebijakan menghapus izinnya dari identitas.
- Ditanamkan – Dipakai dengan kebijakan yang selaras. Anda menyematkan kebijakan inline pada sebuah identitas (pengguna, grup pengguna, atau peran). Menanamkan kebijakan berarti menerapkan izin kebijakan ke identitas. Karena kebijakan inline disimpan di identitas, ia disematkan dan bukan ditempelkan, walau hasilnya serupa.

  > **Note**
  >
  > Anda bisa menanamkan kebijakan inline peran terkait layanan hanya dalam layanan yang bergantung pada peran tersebut. Lihat dokumentasi AWS dari layanan Anda untuk melihat apakah ia mendukung fitur ini.

- Hapus – Dipakai dengan kebijakan yang selaras. Anda menghapus kebijakan sebaris dari IAM identitas (pengguna, grup pengguna, atau peran). Menghapus kebijakan akan menghapus izinnya dari identitas.

  > **Note**
  >
  > Anda bisa menghapus kebijakan inline untukperan terkait layanan hanya dalam layanan yang bergantung pada peran tersebut. Lihat dokumentasi AWS dari layanan Anda untuk melihat apakah ia mendukung fitur ini.

Anda dapat menggunakan konsol, AWS CLI, atau AWS API untuk melakukan salah satu tindakan ini.

Informasi lain

- Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan inline, lihat Kebijakan terkelola dan kebijakan inline.
- Untuk informasi lebih lanjut tentang batasan izin, lihat Batas izin untuk entitas IAM.
- Untuk informasi umum tentang IAM kebijakan, lihatKebijakan dan izin di AWS Identity and Access Management.

- Untuk informasi tentang memvalidasi IAM kebijakan, lihatValidasi kebijakan IAM.
- Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Lihat aktivitas identitas

Sebelum mengubah izin untuk identitas (pengguna, grup pengguna, atau peran), Anda harus melihat aktivitas tingkat layanan mereka yang terkini. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menambahkan izin identitas IAM (konsol)

Anda dapat menggunakan AWS Management Console untuk menambahkan izin ke identitas (pengguna, grup pengguna, atau peran). Untuk melakukan hal ini, tempelkan kebijakan terkelola yang mengendalikan izin, atau tentukan kebijakan yang berfungsi sebagai batasan perizinan. Anda juga bisa menyematkan kebijakan inline.

Untuk menggunakan kebijakan terkelola sebagai kebijakan izin untuk identitas (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih tombol radio di sebelah nama kebijakan yang akan dilampirkan. Anda dapat menggunakan kotak pencarian untuk menyaring daftar kebijakan.
4. Pilih Tindakan, lalu pilih Lampirkan.
5. Pilih satu atau lebih identitas untuk melampirkan kebijakan. Anda bisa memakai kotak pencarian untuk memfilter daftar entitas penanggung jawab. Setelah memilih identitas, pilih Lampirkan kebijakan.

Untuk menggunakan kebijakan terkelola untuk menetapkan batasan izin (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.

3. Dari daftar kebijakan, pilih nama kebijakan yang ingin diatur. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pada halaman detail kebijakan, pilih tab Entitas yang dilampirkan, lalu, jika perlu, buka bagian Terlampir sebagai batas izin dan pilih Tetapkan kebijakan ini sebagai batas izin.
5. Pilih satu atau lebih pengguna atau peran untuk menggunakan kebijakan sebagai batasan izin. Anda bisa memakai kotak pencarian untuk memfilter daftar entitas penanggung jawab. Setelah memilih prinsipal, pilih Setel batas izin.

Untuk menyematkan kebijakan inline bagi pengguna atau peran (konsole)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna atau Peran.
3. Dari daftar, pilih nama pengguna atau peran untuk menanamkan kebijakan ke dalamnya.
4. Pilih tab Izin.
5. Pilih Tambahkan izin, lalu pilih Buat kebijakan sebaris.

> **Note**
>
> Anda tidak dapat menyematkan kebijakan sebaris dalam peran terkait layanan di. IAM Karena layanan tertaut menentukan apakah Anda dapat mengubah izin peran, Anda mungkin dapat menambahkan kebijakan tambahan dari konsol layananAPI, atau. AWS CLI Untuk melihat dokumentasi peran terkait-layanan untuk layanan, lihat AWS layanan yang bekerja dengan IAM dan pilih Ya pada kolom Peran Terkait Layanan untuk layanan Anda.

6. Pilih dari metode berikut untuk melihat langkah-langkah yang diperlukan untuk membuat kebijakan Anda:

   - Mengimpor kebijakan terkelola yang ada – Anda bisa mengimpor kebijakan yang dikelola di dalam akun Anda dan lalu mengedit kebijakan itu untuk menyesuaikan dengan kebutuhan khusus Anda. Kebijakan terkelola dapat berupa kebijakan AWS terkelola atau kebijakan terkelola pelanggan yang Anda buat sebelumnya.
   - Membuat kebijakan dengan editor visual – Anda bisa membuat kebijakan baru dari nol di editor visual. Jika Anda menggunakan editor visual, Anda tidak perlu memahami JSON sintaks.

- Membuat kebijakan menggunakan JSON editor— Dalam opsi JSONeditor, Anda dapat menggunakan JSON sintaks untuk membuat kebijakan. Anda dapat mengetikkan dokumen JSON kebijakan baru atau menempelkan contoh kebijakan.

7. Setelah Anda membuat kebijakan inline, ia akan secara otomatis tertanam di pengguna atau peran Anda.

Untuk menanam kebijakan inline bagi grup pengguna (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih User groups (Grup pengguna).
3. Dari daftar, pilih nama pengguna dari grup pengguna untuk menyematkan kebijakan ke dalamnya.
4. Pilih tab Permissions (Izin), pilih Add permissions (Tambahkan izin), lalu pilih Create inline policy (Membuat kebijakan inline).
5. Lakukan salah satu hal berikut ini:
   - Pilih opsi Visual untuk membuat kebijakan. Untuk informasi selengkapnya, lihat Membuat kebijakan dengan editor visual.
   - Pilih JSONopsi untuk membuat kebijakan. Untuk informasi selengkapnya, lihat Membuat kebijakan menggunakan JSON editor.
6. Jika Anda puas dengan kebijakan ini, pilih Create policy (Buat kebijakan).

Untuk mengubah batasan izin untuk satu atau lebih entitas (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan yang ingin diatur. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pada halaman detail kebijakan, pilih tab Entitas yang dilampirkan, lalu, jika perlu, buka bagian Terlampir sebagai batas izin. Pilih kotak centang di samping pengguna atau peran yang batasnya ingin Anda ubah, lalu pilih Ubah.

5. Pilih kebijakan yang akan dipakai sebagai batasan izin. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan. Setelah memilih kebijakan, pilih Setel batas izin.

Menghapus izin identitas IAM (konsol)

Anda dapat menggunakan AWS Management Console untuk menghapus izin dari identitas (pengguna, grup pengguna, atau peran). Untuk melakukan hal ini, lepaskan kebijakan terkelola yang mengendalikan izin, atau hapus kebijakan yang berfungsi sebagai batasan perizinan. Anda juga bisa menghapus kebijakan inline.

Untuk melepas kebijakan terkelola yang digunakan sebagai kebijakan perizinan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih tombol radio di sebelah nama kebijakan yang akan dilepas. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih Actions (Tindakan), lalu pilih Detach (Lepas).
5. Pilih identitas yang ingin dilepas kebijakannya. Anda bisa memakai kotak pencarian untuk memfilter daftar identitas. Setelah memilih identitas, pilih Lepaskan kebijakan.

Untuk menghapus batasan perizinan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan yang ingin diatur. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pada halaman ringkasan kebijakan, pilih tab Entitas yang dilampirkan, lalu, jika perlu, buka bagian Terlampir sebagai batas izin dan pilih entitas untuk menghapus batas izin. Kemudian pilih Hapus batas.
5. Konfirmasikan bahwa Anda ingin menghapus batas dan memilih Hapus batas.

Untuk menghapus kebijakan inline (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, atauPeran.
3. Dari daftar, pilih nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
4. Pilih tab Izin.
5. Pilih kotak centang di samping kebijakan dan pilih Hapus.
6. Pilih Hapus di kotak konfirmasi.

Menambahkan Kebijakan IAM (AWS CLI)

Anda dapat menggunakan AWS CLI untuk menambahkan izin ke identitas (pengguna, grup pengguna, atau peran). Untuk melakukan hal ini, tempelkan kebijakan terkelola yang mengendalikan izin, atau tentukan kebijakan yang berfungsi sebagai batasan perizinan. Anda juga bisa menyematkan kebijakan inline.

Untuk menggunakan kebijakan terkelola sebagai kebijakan izin untuk entitas (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan
   - Untuk mengambil informasi rinci tentang kebijakan yang dikelola: ambil-kebijakan
2. Untuk melampirkan kebijakan terkelola ke identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
   - aws iam attach-user-policy
   - aws iam attach-group-policy
   - aws iam attach-role-policy

Untuk menggunakan kebijakan terkelola untuk menetapkan batasan perizinan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan

- Untuk mengambil informasi rinci tentang kebijakan yang dikelola: aws-iam ambil-kebijakan

2. Untuk memakai kebijakan yang dikelola pada pengaturan batasan izin untuk entitas (pengguna atau peran), gunakan salah satu perintah berikut:

   - aws iam put-user-permissions-boundary
   - aws iam put-role-permissions-boundary

Untuk menyematkan kebijakan inline (AWS CLI)

Untuk menanamkan kebijakan inline untuk identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), gunakan salah satu perintah berikut:

- aws iam put-user-policy
- aws iam put-group-policy
- aws iam put-role-policy

Menghapus kebijakan IAM (AWS CLI)

Anda dapat menggunakan AWS CLI untuk melepaskan kebijakan terkelola yang mengontrol izin, atau menghapus kebijakan yang berfungsi sebagai batas izin. Anda juga bisa menghapus kebijakan inline.

Untuk melepas kebijakan terkelola yang digunakan sebagai kebijakan perizinan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang kebijakan, jalankan perintah berikut:

   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan
   - Untuk mengambil informasi rinci tentang kebijakan yang dikelola: aws-iam ambil-kebijakan

2. (Opsional) Untuk mengetahui hubungan antara kebijakan dan identitas, jalankan perintah berikut:

   - Untuk mencantumkan identitas (IAMpengguna, IAM grup, dan IAM peran) yang dilampirkan kebijakan terkelola:
     - aws iam list-entities-for-policy
   - Untuk mendaftar kebijakan terkelola yang menempel pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
     - aws iam list-attached-user-policies

- aws iam list-attached-group-policies
- aws iam list-attached-role-policies

3. Untuk melepas kebijakan terkelola dari identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

   - aws iam detach-user-policy
   - aws iam detach-group-policy
   - aws iam detach-role-policy

Untuk menghapus batasan perizinan (AWS CLI)

1. (Opsional) Untuk melihat kebijakan terkelola mana yang saat ini dipakai untuk mengatur batasan izin bagi pengguna atau peran, jalankan perintah berikut:

   - aws iam mendapatkan pengguna
   - aws iam mendapatkan peran

2. (Opsional) Untuk melihat pengguna atau peran dimana kebijakan terkelola dipakai untuk batasan izin, jalankan perintah berikut:

   - aws iam list-entities-for-policy

3. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, jalankan perintah berikut:

   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan
   - Untuk mengambil informasi rinci tentang kebijakan yang dikelola: aws-iam ambil-kebijakan

4. Untuk menghapus batasan izin dari pengguna atau peran, gunakan salah satu perintah berikut:

   - aws iam delete-user-permissions-boundary
   - aws iam delete-role-permissions-boundary

Untuk menghapus kebijakan inline (AWS CLI)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

   - aws iam list-user-policies
   - aws iam list-group-policies

- aws iam list-role-policies

2. (Opsional) Untuk mengambil dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy

3. Untuk menghapus kebijakan inline dari identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), gunakan salah satu perintah berikut:

   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

Menambahkan IAM kebijakan (AWS API)

Anda dapat menggunakan AWS API untuk melampirkan kebijakan terkelola yang mengontrol izin atau menentukan kebijakan yang berfungsi sebagai batas izin. Anda juga bisa menyematkan kebijakan inline.

Untuk menggunakan kebijakan terkelola sebagai kebijakan izin untuk entitas ()AWS API

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:

   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk mengambil informasi rinci tentang kebijakan terkelola: GetPolicy

2. Untuk melampirkan kebijakan terkelola pada sebuah identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:

   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

Untuk menggunakan kebijakan terkelola untuk menetapkan batas izin ()AWS API

1. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, hubungi layanan berikut:

- Untuk mencantumkan kebijakan terkelola: ListPolicies
- Untuk mengambil informasi rinci tentang kebijakan terkelola: GetPolicy

2. Untuk menggunakan kebijakan terkelola pada penetapan batas perizinan bagi suatu entitas (pengguna atau peran), hubungi salah satu layanan berikut:

   - PutUserPermissionsBoundary
   - PutRolePermissionsBoundary

Untuk menyematkan kebijakan inline (AWS API)

Untuk menyimpan kebijakan inline dalam suatu identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), hubungi salah satu layanan berikut:

- PutUserPolicy
- PutGroupPolicy
- PutRolePolicy

Menghapus IAM kebijakan (AWS API)

Anda dapat menggunakan AWS API untuk melepaskan kebijakan terkelola yang mengontrol izin atau menghapus kebijakan yang berfungsi sebagai batas izin. Anda juga bisa menghapus kebijakan inline.

Untuk melepaskan kebijakan terkelola yang digunakan sebagai kebijakan izin ()AWS API

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:

   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk mengambil informasi rinci tentang kebijakan terkelola: GetPolicy

2. (Opsional) Untuk mengetahui perihal hubungan antara kebijakan dan identitas, hubungi layanan berikut:

   - Untuk mencantumkan identitas (IAMpengguna, IAM grup, dan IAM peran) yang dilampirkan kebijakan terkelola:
     - ListEntitiesForPolicy
   - Untuk mencantumkan kebijakan terkelola pada suatu identitas (pengguna, grup pengguna, atau peran), panggil salah satu operasi berikut:

- ListAttachedUserPolicies
- ListAttachedGroupPolicies
- ListAttachedRolePolicies

3. Untuk melepas kebijakan terkelola dari suatu identitas (pengguna, grup pengguna, atau peran), panggil salah satu operasi berikut:

   - DetachUserPolicy
   - DetachGroupPolicy
   - DetachRolePolicy

Untuk menghapus batasan perizinan (AWS API)

1. Untuk melihat kebijakan terkelola yang sedang digunakan pada penetapan batas perizinan bagi seorang pengguna atau peran, hubungi salah satu layanan berikut:

   - GetUser
   - GetRole

2. (Opsional) Untuk melihat pengguna atau peran yang menggunakan kebijakan terkelola pada batas perizinan, hubungi layanan berikut:

   - ListEntitiesForPolicy

3. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, hubungi layanan berikut:

   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk mengambil informasi rinci tentang kebijakan terkelola: GetPolicy

4. Untuk menghilangkan batas perizinan dari pengguna atau peran, hubungi salah satu layanan berikut:

   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

Untuk menghapus kebijakan inline (AWS API)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:

- ListUserPolicies
- ListGroupPolicies
- ListRolePolicies

2. (Opsional) Untuk memperoleh dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

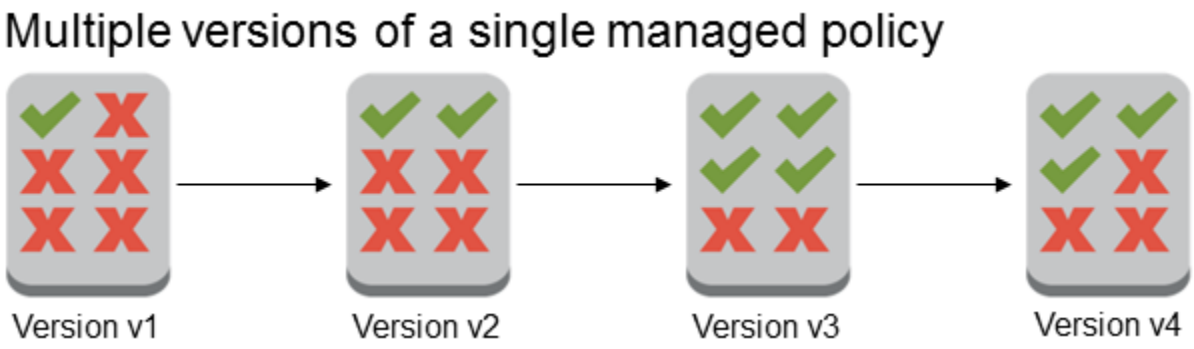
3. Untuk menghapus kebijakan inline dari suatu identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), hubungi salah satu layanan berikut:

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Kebijakan IAM versioning

Saat Anda membuat perubahan pada kebijakan yang dikelola IAM pelanggan, dan saat AWS membuat perubahan pada kebijakan AWS terkelola, kebijakan yang diubah tidak akan menimpa kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola. IAMmenyimpan hingga lima versi kebijakan yang dikelola pelanggan Anda. IAMtidak mendukung pembuatan versi untuk kebijakan sebaris.

Diagram berikut mengilustrasikan versioning untuk kebijakan terkelola pelanggan. Dalam contoh ini, versi 1-4 disimpan. Anda dapat menyimpan hingga lima versi kebijakan terkelolaIAM. Saat mengedit kebijakan yang akan membuat versi tersimpan keenam, Anda dapat memilih versi lama yang tidak akan disimpan lagi. Anda dapat kembali ke salah satu dari empat versi tersimpan lainnya kapan saja.



Versi kebijakan berbeda dengan `Version` elemen kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Untuk mempelajari lebih lanjut tentang `Version` elemen kebijakan lihat IAMJSONelemen kebijakan: Version.

Anda dapat menggunakan beberapa versi untuk melacak perubahan pada kebijakan terkelola. Misalnya, Anda mungkin melakukan perubahan pada kebijakan terkelola dan kemudian menemukan bahwa perubahan tersebut memiliki efek yang tak diinginkan. Dalam kejadian ini, Anda dapat kembali ke kebijakan terkelola versi sebelumnya dengan mengatur versi sebelumnya sebagai versi default.

Topik berikut menjelaskan bagaimana Anda dapat menggunakan pembuatan versi untuk kebijakan terkelola.

Topik

- Pembatasan versi
- Gunakan versi untuk memutar kembali perubahan
- Izin untuk mengatur versi default kebijakan
- Mengatur versi default kebijakan terkelola pelanggan

Pembatasan versi

Kebijakan terkelola dapat memiliki sampai dengan lima versi. Jika Anda perlu membuat perubahan pada kebijakan terkelola di luar lima versi dari AWS Command Line Interface, atau AWS API, Anda harus menghapus satu atau beberapa versi yang ada terlebih dahulu. Jika Anda menggunakan AWS Management Console, Anda tidak perlu menghapus versi sebelum mengedit kebijakan Anda. Ketika Anda menyimpan versi keenam, muncul kotak dialog yang menyarankan Anda menghapus satu atau beberapa versi non-default kebijakan Anda. Anda dapat melihat dokumen JSON kebijakan untuk setiap versi untuk membantu Anda memutuskan. Untuk detail tentang kotak dialog ini, lihat the section called “Edit IAM kebijakan”.

Anda dapat menghapus versi kebijakan terkelola apa pun yang Anda inginkan, kecuali versi default. Ketika Anda menghapus suatu versi, pengidentifikasi versi untuk sisanya tidak berubah. Hasilnya, pengidentifikasi versi mungkin tak berurutan. Sebagai contoh, apabila Anda menghapus kebijakan terkelola versi v2 dan v4 dan menambahkan dua versi baru, pengidentifikasi versi lainnya mungkin adalah v1, v3, v5, v6, dan v7.

Gunakan versi untuk memutar kembali perubahan

Anda dapat mengatur versi default kebijakan terkelola pelanggan untuk mengembalikan perubahan Anda. Sebagai contoh, pertimbangkan alur perencanaan berikut ini:

Anda membuat kebijakan terkelola pelanggan yang memungkinkan pengguna mengelola bucket Amazon S3 tertentu menggunakan. AWS Management Console Setelah pembuatan, kebijakan terkelola pelanggan Anda hanya memiliki satu versi, yang diidentifikasi sebagai v1, sehingga versi tersebut secara otomatis diatur sebagai default. Kebijakan ini bekerja sebagaimana mestinya.

Kemudian, Anda memperbaharui kebijakan untuk menambahkan izin mengelola bucket Amazon S3 kedua. IAMmembuat versi baru kebijakan, diidentifikasi sebagai v2, yang berisi perubahan Anda. Anda mengatur versi v2 sebagai standar, dan tidak lama kemudian pengguna Anda melaporkan bahwa mereka tidak memiliki izin untuk menggunakan konsol Amazon S3. Dalam hal ini, Anda dapat kembali ke kebijakan versi v1, yang Anda tahu bekerja sebagaimana mestinya. Untuk melakukannya, Anda menetapkan versi v1 sebagai versi standar. Pengguna Anda sekarang dapat menggunakan konsol Amazon S3 untuk mengelola bucket asli.

Kemudian, setelah Anda menentukan kesalahan dalam kebijakan versi v2, Anda memperbaharui kebijakannya lagi untuk menambahkan izin mengelola bucket Amazon S3 yang kedua. IAMmembuat versi baru kebijakan lainnya, yang diidentifikasi sebagai v3. Anda mengatur versi v3 sebagai default, dan versi ini berjalan sebagaimana mestinya. Pada tahap ini, Anda menghapus versi v2 kebijakan.

Izin untuk mengatur versi default kebijakan

Izin yang diperlukan untuk menyetel versi default kebijakan sesuai dengan AWS API operasi untuk tugas tersebut. Anda dapat menggunakan `CreatePolicyVersion` atau `SetDefaultPolicyVersion` API operasi untuk menyetel versi default kebijakan. Untuk memperbolehkan seseorang mengatur versi kebijakan default dari kebijakan yang ada, Anda dapat mengizinkan akses untuk tindakan `iam:CreatePolicyVersion` atau tindakan `iam:SetDefaultPolicyVersion`. Tindakan `iam:CreatePolicyVersion` memungkinkan mereka membuat kebijakan versi baru dan mengatur versi tersebut sebagai default. Tindakan

`iam:SetDefaultPolicyVersion` memungkinkan mereka mengatur versi kebijakan yang ada sebagai kebijakan default.

> **Important**
>
> Menolak tindakan `iam:SetDefaultPolicyVersion` dalam kebijakan pengguna tidak menghentikan pengguna untuk membuat versi kebijakan baru dan mengaturnya sebagai default.

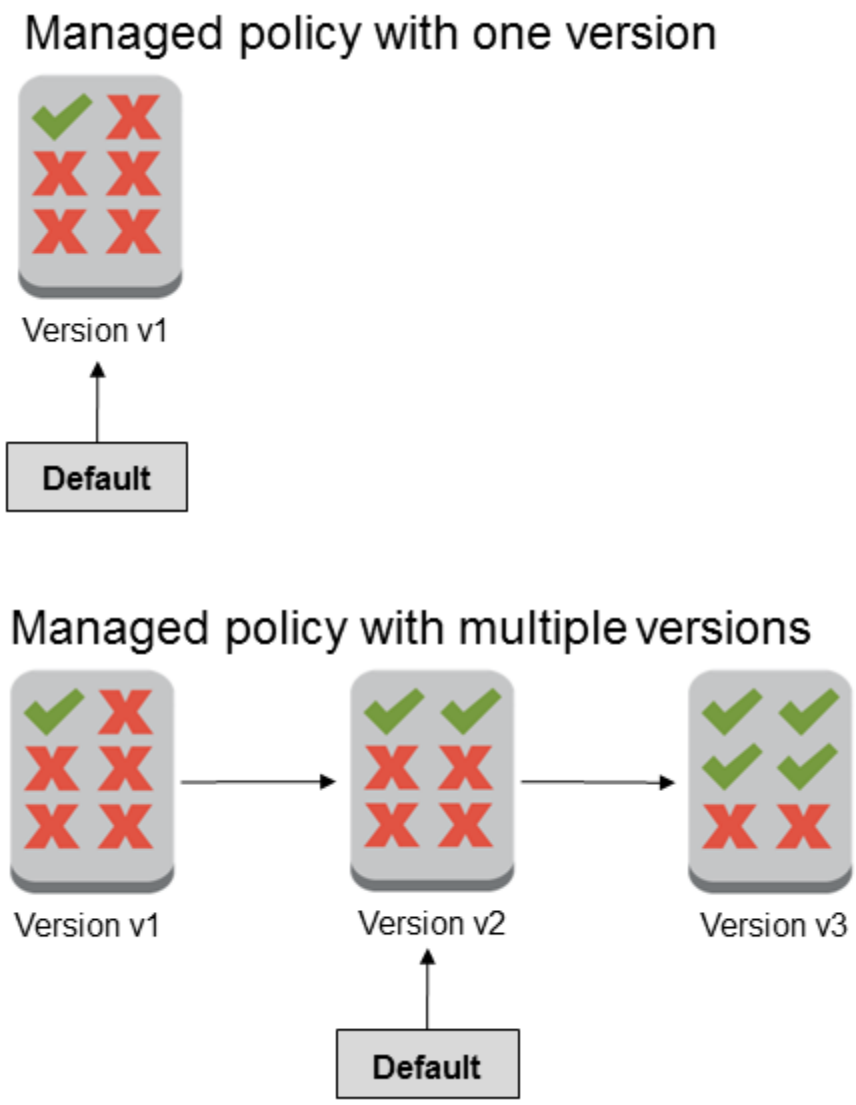
Anda dapat menggunakan kebijakan berikut untuk menolak akses pengguna untuk mengubah kebijakan terkelola pelanggan yang sudah ada:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
}
```

Mengatur versi default kebijakan terkelola pelanggan

Salah satu versi kebijakan terkelola ditetapkan sebagai versi default. Versi default kebijakan adalah versi operatif—yaitu, ini adalah versi yang berlaku untuk semua entitas utama (IAMpengguna, IAM grup, dan IAM peran) tempat kebijakan terkelola dilampirkan.

Ketika Anda membuat kebijakan terkelola pelanggan, kebijakan tersebut dimulai versi tunggal yang diidentifikasi sebagai v1. Untuk kebijakan terkelola dengan versi tunggal saja, versi tersebut secara otomatis diatur sebagai default. Untuk kebijakan terkelola pelanggan dengan lebih dari satu versi, Anda memilih versi mana yang akan ditetapkan sebagai default. Untuk kebijakan AWS terkelola, versi default ditetapkan oleh AWS. Diagram berikut mengilustrasikan hal ini.



Anda dapat menyetel versi default kebijakan terkelola pelanggan untuk menerapkan versi tersebut ke setiap IAM identitas (pengguna, grup pengguna, dan peran) tempat kebijakan dilampirkan. Anda tidak dapat menyetel versi default untuk kebijakan AWS terkelola atau kebijakan sebaris.

Untuk mengatur versi default kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan untuk diatur menjadi versi default. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.

4. Pilih tab Versi kebijakan. Centang kotak di samping versi yang ingin Anda tetapkan sebagai versi default, lalu pilih Atur sebagai default.

Untuk mempelajari cara menyetel versi default kebijakan terkelola pelanggan dari AWS Command Line Interface atau kebijakan AWS API, lihatEdit IAM kebijakan (AWS CLI).

Edit IAM kebijakan

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Kebijakan disimpan AWS sebagai JSON dokumen dan dilampirkan pada prinsipal sebagai kebijakan berbasis identitas di. IAM Anda dapat melampirkan kebijakan berbasis identitas ke prinsipal (atau identitas), seperti grup IAM pengguna, pengguna, atau peran. Kebijakan berbasis identitas mencakup kebijakan terkelola, kebijakan yang AWS dikelola pelanggan, dan kebijakan inline. Anda dapat mengedit kebijakan terkelola pelanggan dan kebijakan inline diIAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Umumnya lebih baik menggunakan kebijakan yang dikelola pelanggan daripada kebijakan inline atau kebijakan AWS terkelola. AWS kebijakan terkelola biasanya memberikan izin administratif atau hanya-baca yang luas. Kebijakan inline tidak dapat digunakan kembali pada identitas lain atau dikelola di luar identitas di mana mereka ada. Untuk keamanan terbesar, berikan hak istimewa paling sedikit, yang berarti hanya memberikan izin yang diperlukan untuk melakukan tugas pekerjaan tertentu.

Ketika Anda membuat atau mengedit IAM kebijakan, secara otomatis AWS dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif dengan sedikit hak istimewa dalam pikiran. Dalam AWS Management Console, IAM mengidentifikasi kesalahan JSON sintaks, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat Validasi kebijakan IAMAccess Analyzer.

Anda dapat menggunakan AWS Management Console, AWS CLI, atau AWS API untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline diIAM. Untuk informasi selengkapnya tentang menggunakan AWS CloudFormation templat untuk menambah atau memperbarui kebijakan, lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan AWS CloudFormation Pengguna.

Topik

Edit IAM kebijakan (konsol)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS Management Console untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline diIAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan IAMJSONreferensi elemen kebijakan.

Prasyarat

Sebelum Anda mengubah izin untuk suatu kebijakan, Anda harus meninjau aktivitas tingkat layanannya yang terbaru. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menyunting kebijakan terkelola pelanggan (konsol)

Anda dapat mengedit kebijakan terkelola pelanggan untuk mengubah izin yang ditentukan dalam kebijakan dari AWS Management Console. Kebijakan terkelola pelanggan dapat memiliki sampai dengan lima versi. Hal ini penting karena apabila Anda melakukan perubahan pada kebijakan terkelola lebih dari lima versi, AWS Management Console akan menyarankan agar Anda memutuskan versi mana yang akan dihapus. Anda juga dapat mengubah versi default atau menghapus suatu versi kebijakan sebelum Anda menyuntingnya untuk menghindari anjuran. Untuk mempelajari lebih lanjut tentang versi, lihat Kebijakan IAM versioning.

Untuk menyunting kebijakan yang dikelola pelanggan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan untuk disunting. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Izin, lalu pilih Edit.
5. Lakukan salah satu hal berikut ini:

   - Pilih opsi Visual untuk mengubah kebijakan Anda tanpa memahami JSON sintaks. Anda dapat membuat perubahan pada layanan, tindakan, sumber daya, atau kondisi opsional untuk setiap blokir izin dalam kebijakan Anda. Anda juga dapat mengimpor kebijakan untuk menambahkan izin tambahan ke bawah kebijakan Anda. Setelah selesai melakukan perubahan, pilih Berikutnya untuk melanjutkan.
   - Pilih JSONopsi untuk mengubah kebijakan Anda dengan mengetik atau menempelkan teks di kotak JSON teks. Anda juga dapat mengimpor kebijakan untuk menambahkan izin tambahan ke bawah kebijakan Anda. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

     > **Note**
     >
     > Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

6. Pada halaman Tinjau dan simpan, tinjau Izin yang ditentukan dalam kebijakan ini, lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.
7. Jika kebijakan terkelola sudah memiliki maksimal lima versi, memilih Simpan perubahan akan menampilkan kotak dialog. Untuk menyimpan versi baru Anda, versi kebijakan non-default tertua akan dihapus dan diganti dengan versi baru ini. Secara opsional, Anda dapat mengatur versi baru sebagai versi kebijakan default.

   Pilih Simpan perubahan untuk menyimpan versi kebijakan baru Anda.

Menyetel versi default kebijakan terkelola pelanggan (konsol)

Anda dapat menyetel versi default kebijakan terkelola pelanggan dari AWS Management Console.

Untuk mengatur versi default kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan untuk diatur menjadi versi default. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Versi kebijakan. Centang kotak di samping versi yang ingin Anda tetapkan sebagai versi default, lalu pilih Atur sebagai default.

Menghapus versi kebijakan terkelola pelanggan (konsol)

Anda dapat menghapus versi kebijakan yang dikelola pelanggan dari AWS Management Console.

Untuk menghapus versi kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Pilih nama kebijakan terkelola pelanggan dengan versi yang ingin Anda hapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Versi kebijakan. Centang kotak di samping versi yang ingin Anda hapus. Kemudian pilih Hapus.
5. Konfirmasi bahwa Anda ingin menghapus versi tersebut, lalu pilih Hapus.

Mengedit kebijakan inline (konsol)

Anda dapat mengedit kebijakan inline dari AWS Management Console.

Untuk mengedit kebijakan inline bagi pengguna, grup pengguna, atau peran (konsol)

1. Di panel navigasi, pilih Pengguna, Grup Pengguna, atauPeran.
2. Dari daftar, pilih nama pengguna, grup pengguna, atau peran dengan kebijakan yang ingin Anda modifikasi. Lalu pilih tabPerizinan dan kembangkan kebijakan.
3. Untuk mengedit kebijakan inline, pilih Edit Policy (Edit Kebijakan).
4. Lakukan salah satu hal berikut ini:

- Pilih opsi Visual untuk mengubah kebijakan Anda tanpa memahami JSON sintaks. Anda dapat membuat perubahan pada layanan, tindakan, sumber daya, atau kondisi opsional untuk setiap blokir izin dalam kebijakan Anda. Anda juga dapat mengimpor kebijakan untuk menambahkan izin tambahan ke bawah kebijakan Anda. Setelah selesai melakukan perubahan, pilih Berikutnya untuk melanjutkan.
- Pilih JSONopsi untuk mengubah kebijakan Anda dengan mengetik atau menempelkan teks di kotak JSON teks. Anda juga dapat mengimpor kebijakan untuk menambahkan izin tambahan ke bawah kebijakan Anda. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya. Untuk menyimpan perubahan Anda tanpa memengaruhi entitas yang terlampir sekarang, kosongkan kotak centang untuk Simpan sebagai versi standar.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

5. Pada halaman Tinjauan, tinjau ringkasan kebijakan, lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Edit IAM kebijakan (AWS CLI)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline diIAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan IAMJSONreferensi elemen kebijakan.

Prasyarat

Sebelum Anda mengubah izin untuk suatu kebijakan, Anda harus meninjau aktivitas tingkat layanannya yang terbaru. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal

(orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Mengedit kebijakan terkelola pelanggan (AWS CLI)

Anda dapat mengedit kebijakan yang dikelola pelanggan dari AWS CLI.

> **Note**
>
> Kebijakan terkelola dapat memiliki hingga lima versi. Apabila Anda perlu melakukan perubahan pada kebijakan terkelola pelanggan lebih dari lima versi, Anda harus terlebih dahulu menghapus satu atau beberapa versi yang ada.

Untuk mengedit kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang suatu kebijakan, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: daftar-kebijakan
   - Untuk memperoleh informasi terperinci tentang kebijakan terkelola: dapatkan-kebijakan
2. (Opsional) Untuk mengetahui tentang hubungan antara kebijakan dan identitas, jalankan perintah berikut:
   - Untuk mencantumkan identitas (IAMpengguna, IAM grup, dan IAM peran) yang dilampirkan kebijakan terkelola:
     - list-entities-for-policy
   - Untuk membuat daftar kebijakan terkelola yang terlampir pada identitas (pengguna, grup pengguna, atau peran):
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Untuk menyunting kebijakan terkelola pelanggan, jalankan perintah berikut:
   - create-policy-version
4. (Opsional) Untuk memvalidasi kebijakan terkelola pelanggan, jalankan perintah IAM Access Analyzer berikut:

- validasi-kebijakan

Menyetel versi default dari kebijakan terkelola pelanggan (AWS CLI)

Anda dapat menyetel versi default kebijakan terkelola pelanggan dari AWS CLI.

Untuk mengatur versi default kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, jalankan perintah berikut:
   - daftar-kebijakan
2. Untuk mengatur versi default kebijakan terkelola pelanggan, jalankan perintah berikut:
   - set-default-policy-version

Menghapus versi kebijakan terkelola pelanggan ()AWS CLI

Anda dapat menghapus versi kebijakan yang dikelola pelanggan dari AWS CLI.

Untuk menghapus versi kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, jalankan perintah berikut:
   - daftar-kebijakan
2. Untuk menghapus kebijakan terkelola pelanggan, jalankan perintah berikut:
   - delete-policy-version

Mengedit kebijakan sebaris ()AWS CLI

Anda dapat mengedit kebijakan inline dari AWS CLI.

Untuk mengedit kebijakan inline ()AWS CLI

1. (Opsional) Untuk melihat informasi tentang suatu kebijakan, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan sebaris yang terkait dengan identitas (pengguna, grup pengguna, atau peran):
     - list-user-policies

    - list-role-policies
    - list-group-policies
  - Untuk mengambil informasi rinci tentang kebijakan inline:
    - get-user-policy
    - get-role-policy
    - get-group-policy
2. Untuk mengedit kebijakan inline, jalankan perintah berikut:
    - put-user-policy
    - put-role-policy
    - put-group-policy
3. (Opsional) Untuk memvalidasi kebijakan inline, jalankan perintah IAM Access Analyzer berikut:
    - validasi-kebijakan

Edit IAM kebijakan (AWS API)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS API untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline diIAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan IAMJSONreferensi elemen kebijakan.

Prasyarat

Sebelum Anda mengubah izin untuk suatu kebijakan, Anda harus meninjau aktivitas tingkat layanannya yang terbaru. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Mengedit kebijakan terkelola pelanggan (AWS API)

Anda dapat mengedit kebijakan yang dikelola pelanggan menggunakan AWS API.

> **Note**
>
> Kebijakan terkelola dapat memiliki hingga lima versi. Apabila Anda perlu melakukan perubahan pada kebijakan terkelola pelanggan lebih dari lima versi, Anda harus terlebih dahulu menghapus satu atau beberapa versi yang ada.

Untuk mengedit kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:
   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk mengambil informasi rinci tentang kebijakan terkelola: GetPolicy
2. (Opsional) Untuk mengetahui perihal hubungan antara kebijakan dan identitas, hubungi layanan berikut:
   - Untuk mencantumkan identitas (IAMpengguna, IAM grup, dan IAM peran) yang dilampirkan kebijakan terkelola:
     - ListEntitiesForPolicy
   - Untuk membuat daftar kebijakan terkelola yang terlampir pada identitas (pengguna, grup pengguna, atau peran):
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Untuk menyunting kebijakan terkelola pelanggan, hubungi layanan berikut:
   - CreatePolicyVersion
4. (Opsional) Untuk memvalidasi kebijakan terkelola pelanggan, hubungi operasi IAM Access Analyzer berikut:
   - ValidatePolicy

Menyetel versi default kebijakan terkelola pelanggan (AWS API)

Anda dapat menyetel versi default kebijakan terkelola pelanggan dari AWS API.

Untuk menyetel versi default kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, hubungi layanan berikut:
   - ListPolicies
2. Untuk mengatur versi standar kebijakan terkelola pelanggan, hubungi layanan berikut:
   - SetDefaultPolicyVersion

Menghapus versi kebijakan terkelola pelanggan ()AWS API

Anda dapat menghapus versi kebijakan yang dikelola pelanggan dari AWS API.

Untuk menghapus versi kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, hubungi layanan berikut:
   - ListPolicies
2. Untuk menghapus kebijakan terkelola pelanggan, hubungi layanan berikut:
   - DeletePolicyVersion

Mengedit kebijakan sebaris ()AWS API

Anda dapat mengedit kebijakan inline dari. AWS API

Untuk mengedit kebijakan inline ()AWS API

1. (Opsional) Untuk melihat informasi tentang kebijakan sebaris, jalankan operasi berikut:
   - Untuk mencantumkan kebijakan sebaris yang terkait dengan identitas (pengguna, grup pengguna, atau peran):
     - ListUserPolicies
     - ListRolePolicies
     - ListGroupPolicies
   - Untuk mengambil informasi rinci tentang kebijakan inline:
     - GetUserPolicy
     - GetRolePolicy

- GetGroupPolicy

2. Untuk mengedit kebijakan inline, jalankan operasi berikut:

   - PutUserPolicy
   - PutRolePolicy
   - PutGroupPolicy

3. (Opsional) Untuk memvalidasi kebijakan inline, jalankan operasi IAM Access Analyzer berikut:

   - ValidatePolicy

Hapus IAM kebijakan

Anda dapat menghapus IAM kebijakan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API.

> **Note**
>
> Penghapusan IAM kebijakan bersifat permanen. Setelah kebijakan dihapus, kebijakan tersebut tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur IAM kebijakan dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan. IAMJSONreferensi elemen kebijakan

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

Topik

- Hapus IAM kebijakan (konsol)
- Hapus IAM kebijakan (AWS CLI)
- Hapus IAM kebijakan (AWS API)

Hapus IAM kebijakan (konsol)

Anda dapat menggunakan AWS Management Console untuk menghapus kebijakan terkelola pelanggan dan kebijakan inline diIAM. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

> **Note**
>
> Penghapusan IAM kebijakan bersifat permanen. Setelah kebijakan dihapus, kebijakan tersebut tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur IAM kebijakan dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan. IAMJSONreferensi elemen kebijakan

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Prasyarat

Sebelum menghapus suatu kebijakan, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus kebijakan IAM (konsol)

Anda dapat menghapus kebijakan yang dikelola pelanggan untuk menghapusnya dari kebijakan Anda Akun AWS. Anda tidak dapat menghapus kebijakan AWS terkelola.

Untuk menyunting kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Pilih tombol radio di sebelah kebijakan terkelola pelanggan untuk dihapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih Tindakan, lalu pilih Hapus.

5. Ikuti petunjuk untuk mengonfirmasi bahwa Anda ingin menghapus kebijakan, lalu pilih Hapus.

Menghapus kebijakan sebaris (konsol)

Anda dapat menghapus kebijakan inline untuk menghapusnya dari kebijakan Anda Akun AWS. Anda tidak dapat menghapus kebijakan terkelola AWS .

Untuk menghapus kebijakan inline bagi grup pengguna, pengguna, atau peran (konsol)

1. Di panel navigasi, pilih User groups (Grup pengguna), Users (Pengguna), atau Roles (Peran).
2. Pilih nama grup pengguna, pengguna, atau peran dengan kebijakan yang ingin Anda hapus. Kemudian pilih tab Perizinan.
3. Pilih kotak centang di samping kebijakan yang akan dihapus dan pilih Hapus. Untuk menghapus kebijakan sebaris di Pengguna atau Peran, pilih Hapus untuk mengonfirmasi penghapusan. Jika Anda menghapus kebijakan inline tunggal di User groups (Pengguna grup), ketik nama kebijakan dan pilih Delete (Hapus). Jika Anda menghapus beberapa kebijakan inline diPengguna grup, ketik jumlah kebijakan yang Anda hapus diikuti oleh **`inline policies`**, lalu pilih Delete (Hapus). Sebagai contoh, jika Anda menghapus tiga kebijakan inline, ketik **`3 inline policies`**.

Hapus IAM kebijakan (AWS CLI)

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk menghapus kebijakan terkelola pelanggan dan kebijakan inline diIAM. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

> **Note**
>
> Penghapusan IAM kebijakan bersifat permanen. Setelah kebijakan dihapus, kebijakan tersebut tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur IAM kebijakan dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan. IAMJSONreferensi elemen kebijakan

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Prasyarat

Sebelum menghapus suatu kebijakan, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus kebijakan terkelola pelanggan ()AWS CLI

Anda dapat menghapus kebijakan terkelola pelanggan dari AWS Command Line Interface.

Untuk menghapus kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang suatu kebijakan, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: daftar-kebijakan
   - Untuk memperoleh informasi terperinci tentang kebijakan terkelola: dapatkan-kebijakan
2. (Opsional) Untuk mengetahui tentang hubungan antara kebijakan dan identitas, jalankan perintah berikut:
   - Untuk mencantumkan identitas (IAMpengguna, IAM grup, dan IAM peran) tempat kebijakan terkelola dilampirkan, jalankan perintah berikut:
     - list-entities-for-policy
   - Untuk mencantumkan kebijakan terkelola yang terlampir pada sebuah identitas (pengguna, grup pengguna, atau peran), jalankan salah satu perintah berikut:
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Untuk menghapus kebijakan terkelola pelanggan, jalankan perintah berikut:
   - menghapus-kebijakan

Menghapus kebijakan inline ()AWS CLI

Anda dapat menghapus kebijakan inline dari. AWS CLI

Untuk menghapus kebijakan inline (AWS CLI)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies
2. (Opsional) Untuk mengambil dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy
3. Untuk menghapus kebijakan inline dari identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), gunakan salah satu perintah berikut:
   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

Hapus IAM kebijakan (AWS API)

Anda dapat menggunakan AWS API untuk menghapus kebijakan terkelola pelanggan dan kebijakan inline diIAM. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

> **Note**
>
> Penghapusan IAM kebijakan bersifat permanen. Setelah kebijakan dihapus, kebijakan tersebut tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur IAM kebijakan dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan. IAMJSONreferensi elemen kebijakan

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Prasyarat

Sebelum menghapus suatu kebijakan, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus kebijakan terkelola pelanggan ()AWS API

Anda dapat menghapus kebijakan yang dikelola pelanggan menggunakan AWS API.

Untuk menghapus kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:
   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk mengambil informasi rinci tentang kebijakan terkelola: GetPolicy
2. (Opsional) Untuk mengetahui perihal hubungan antara kebijakan dan identitas, hubungi layanan berikut:
   - Untuk mencantumkan identitas (IAMpengguna, IAM grup, dan IAM peran) tempat kebijakan terkelola dilampirkan, panggil operasi berikut:
     - ListEntitiesForPolicy
   - Untuk mencantumkan kebijakan terkelola pada suatu identitas (pengguna, grup pengguna, atau peran), panggil salah satu operasi berikut:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Untuk menghapus kebijakan terkelola pelanggan, hubungi layanan berikut:
   - DeletePolicy

Menghapus kebijakan inline ()AWS API

Anda dapat menghapus kebijakan inline menggunakan. AWS API

Untuk menghapus kebijakan inline (AWS API)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Opsional) Untuk memperoleh dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:
   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy
3. Untuk menghapus kebijakan inline dari suatu identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), hubungi salah satu layanan berikut:
   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses

Sebagai administrator, Anda dapat memberikan izin ke IAM sumber daya (peran, pengguna, grup pengguna, atau kebijakan) di luar yang mereka butuhkan. IAMmemberikan informasi yang terakhir diakses untuk membantu Anda mengidentifikasi izin yang tidak digunakan sehingga Anda dapat menghapusnya. Anda dapat menggunakan informasi yang diakses terakhir untuk menyempurnakan kebijakan Anda dan mengizinkan akses hanya ke layanan dan tindakan yang digunakan IAM identitas dan kebijakan Anda. Ini membantu Anda untuk lebih menganut praktik terbaik dengan privilese paling sedikit. Anda dapat melihat informasi yang terakhir diakses untuk identitas atau kebijakan yang ada di IAM atau AWS Organizations.

Anda dapat terus memantau informasi yang diakses terakhir dengan penganalisis akses yang tidak digunakan. Untuk informasi selengkapnya, lihat Temuan untuk akses eksternal dan tidak terpakai.

Topik

- Jenis informasi yang terakhir diakses untuk IAM

- Informasi yang terakhir diakses untuk AWS Organizations
- Hal yang perlu diketahui tentang informasi yang terakhir diakses
- Izin diperlukan
- Memecahkan masalah aktivitas untuk dan entitas IAM Organizations
- Tempat AWS melacak informasi terakhir yang diakses
- Lihat informasi yang terakhir diakses untuk IAM
- Lihat informasi yang terakhir diakses untuk Organizations
- Contoh alur perencanaan untuk menggunakan informasi yang terakhir diakses
- IAMtindakan terakhir diakses layanan informasi dan tindakan

Jenis informasi yang terakhir diakses untuk IAM

Anda dapat melihat dua jenis informasi IAM identitas yang terakhir diakses: informasi AWS layanan yang diizinkan dan informasi tindakan yang diizinkan. Informasi tersebut mencakup tanggal dan waktu ketika upaya untuk mengakses AWS API dilakukan. Untuk tindakan, informasi yang terakhir diakses melaporkan tindakan manajemen layanan. Tindakan manajemen meliputi pembuatan, penghapusan, dan tindakan modifikasi. Untuk mempelajari lebih lanjut tentang cara melihat informasi yang terakhir diaksesIAM, lihatLihat informasi yang terakhir diakses untuk IAM.

Misalnya skenario untuk menggunakan informasi yang diakses terakhir untuk membuat keputusan tentang izin yang Anda berikan pada IAM identitas Anda, lihat. Contoh alur perencanaan untuk menggunakan informasi yang terakhir diakses

Untuk mempelajari lebih lanjut tentang bagaimana tersedianya informasi untuk tindakan manajemen, lihat Hal yang perlu diketahui tentang informasi yang terakhir diakses.

Informasi yang terakhir diakses untuk AWS Organizations

Jika Anda masuk menggunakan kredensi akun manajemen, Anda dapat melihat informasi layanan yang terakhir diakses untuk AWS Organizations entitas atau kebijakan di organisasi Anda. AWS Organizations entitas termasuk akar organisasi, unit organisasi (OUs), atau akun. Informasi terakhir yang diakses untuk AWS Organizations mencakup informasi tentang layanan yang diizinkan oleh kebijakan kontrol layanan (SCP). Informasi menunjukkan prinsip mana (pengguna root, IAM pengguna, atau peran) dalam organisasi atau akun yang terakhir mencoba mengakses layanan dan kapan. Untuk mempelajari lebih lanjut tentang laporan dan cara melihat informasi yang terakhir diakses AWS Organizations, lihatLihat informasi yang terakhir diakses untuk Organizations.

Sebagai contoh, alur perencanaan penggunaan informasi yang terakhir diakses untuk membuat keputusan tentang izin yang Anda berikan ke entitas organisasi Anda, lihat Contoh alur perencanaan untuk menggunakan informasi yang terakhir diakses.

Hal yang perlu diketahui tentang informasi yang terakhir diakses

Sebelum Anda menggunakan informasi yang terakhir diakses dari laporan untuk mengubah izin IAM identitas atau entitas Organizations, tinjau detail berikut tentang informasi tersebut.

- Periode pelacakan — Aktivitas terbaru muncul di IAM konsol dalam waktu empat jam. Periode pelacakan untuk informasi layanan setidaknya 400 hari tergantung pada kapan layanan mulai melacak informasi tindakan. Periode pelacakan untuk informasi tindakan Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk Amazon EC2IAM,, dan tindakan Lambda dimulai pada 7 April 2021. Periode pelacakan untuk semua layanan lainnya dimulai pada 23 Mei 2023. Untuk daftar layanan yang informasi terakhir diakses tindakan yang tersedia, lihatIAMtindakan terakhir diakses layanan informasi dan tindakan. Untuk informasi selengkapnya tentang tindakan Wilayah mana yang terakhir diakses informasi tersedia, lihatTempat AWS melacak informasi terakhir yang diakses.
- Upaya dilaporkan — Layanan data terakhir diakses mencakup semua upaya untuk mengakses AWS API, bukan hanya upaya yang berhasil. Ini termasuk semua upaya yang dilakukan menggunakan AWS Management Console, AWS API melalui salah satuSDKs, atau salah satu alat baris perintah. Entri tak terduga dalam layanan data terakhir yang diakses tidak berarti akun Anda telah disusupi, karena permintaan tersebut mungkin ditolak. Lihat CloudTrail log Anda sebagai sumber otoritatif untuk informasi tentang semua API panggilan dan apakah mereka berhasil atau ditolak aksesnya.
- PassRole— `iam:PassRole` Tindakan tidak dilacak dan tidak termasuk dalam IAM tindakan informasi yang diakses terakhir.
- Action last access information — Action last access information tersedia untuk tindakan manajemen layanan yang diakses oleh IAM identitas. Lihat daftar layanan dan tindakan mereka untuk tindakan yang terakhir diakses melaporkan informasi.

  > **Note**
  >
  > Informasi tindakan yang terakhir diakses tidak tersedia untuk peristiwa data Amazon S3.

- Peristiwa manajemen - IAM menyediakan informasi tindakan untuk peristiwa manajemen layanan yang dicatat oleh CloudTrail. Terkadang, peristiwa CloudTrail manajemen juga disebut operasi

pesawat kontrol atau peristiwa pesawat kontrol. Acara manajemen memberikan visibilitas ke dalam operasi administratif yang dilakukan pada sumber daya di Anda Akun AWS. Untuk mempelajari selengkapnya tentang peristiwa manajemen di CloudTrail, lihat peristiwa pengelolaan log di Panduan AWS CloudTrail Pengguna.

- Laporkan pemilik – Hanya penanggung jawab yang membuat laporan yang dapat melihat detail laporan. Ini berarti bahwa ketika Anda melihat informasi di AWS Management Console, Anda mungkin harus menunggu untuk menghasilkan dan memuat. Jika Anda menggunakan AWS CLI atau AWS API untuk mendapatkan detail laporan, kredensional Anda harus sesuai dengan kredensi prinsipal yang menghasilkan laporan. Apabila Anda menggunakan kredensial sementara untuk peran atau pengguna gabungan, Anda harus membuat dan mengambil laporan selama sesi yang sama. Untuk informasi lebih lanjut tentang penanggung jawab sesi peran yang diasumsikan, lihat AWS JSONelemen kebijakan: Principal.
- IAMsumber daya — Informasi yang terakhir diakses untuk IAM mencakup IAM sumber daya (peran, pengguna, IAM grup, dan kebijakan) di akun Anda. Informasi terakhir yang diakses untuk Organizations mencakup prinsipal (IAMpengguna, IAM peran, atau Pengguna root akun AWS) dalam entitas Organizations tertentu. Informasi terakhir yang diakses tidak termasuk upaya yang tidak diautentikasi.
- IAMJenis kebijakan — Informasi terakhir yang diakses untuk IAM mencakup layanan yang diizinkan oleh kebijakan IAM identitas. Ini merupakan kebijakan yang melekat pada suatu peran atau terlampir pada pengguna secara langsung atau melalui grup. Akses yang diperbolehkan oleh jenis kebijakan lain tidak termasuk dalam laporan Anda. Jenis kebijakan yang dikecualikan mencakup kebijakan berbasis sumber daya, daftar kontrol akses, batasan IAM izin AWS Organizations SCPs, dan kebijakan sesi. Izin yang disediakan oleh peran terkait layanan ditentukan oleh layanan yang ditautkan dan tidak dapat dimodifikasi. IAM Untuk mempelajari lebih lanjut tentang peran terkait layanan, lihat Buat peran tertaut layanan Untuk mempelajari cara jenis kebijakan yang berbeda dievaluasi untuk memungkinkan atau menolak akses, lihat Logika evaluasi kebijakan.
- Jenis kebijakan Organizations — Informasi untuk hanya AWS Organizations mencakup layanan yang diizinkan oleh kebijakan kontrol layanan warisan entitas Organizations (SCPs). SCPsadalah kebijakan yang dilampirkan ke root, OU, atau akun. Akses yang diperbolehkan oleh jenis kebijakan lain tidak termasuk dalam laporan Anda. Jenis kebijakan yang dikecualikan mencakup kebijakan berbasis identitas, kebijakan berbasis sumber daya, daftar kontrol akses, batas izin, dan kebijakan sesi. IAM Untuk mempelajari cara berbagai jenis kebijakan dievaluasi untuk memungkinkan atau menolak akses, lihat Logika evaluasi kebijakan.

- Menentukan ID kebijakan — Saat Anda menggunakan AWS CLI atau membuat laporan AWS API untuk informasi yang terakhir diakses di Organizations, Anda dapat menentukan ID kebijakan secara opsional. Laporan yang dihasilkan mencakup informasi untuk layanan yang diperbolehkan hanya oleh kebijakan tersebut. Informasi ini mencakup aktivitas akun terbaru di entitas Organisasi yang spesifik atau anak dari entitas tersebut. Untuk informasi selengkapnya, lihat aws iam generate-organizations-access -report atau. GenerateOrganizationsAccessReport
- Akun manajemen organisasi – Anda harus masuk ke akun manajemen organisasi Anda untuk melihat informasi yang terakhir diakses oleh layanan. Anda dapat memilih untuk melihat informasi untuk akun manajemen menggunakan IAM konsol, AWS CLI, atau AWS API. Laporan yang dihasilkan mencantumkan semua AWS layanan, karena akun manajemen tidak dibatasi olehSCPs. Jika Anda menentukan ID kebijakan di CLI atauAPI, kebijakan akan diabaikan. Untuk setiap layanan, laporan mencakup informasi hanya untuk akun manajemen. Namun, laporan untuk entitas Organisasi lain tidak mengembalikan informasi atas aktivitas di akun manajemen.
- Pengaturan Organizations — Administrator harus mengaktifkan SCPs di root organisasi Anda sebelum Anda dapat menghasilkan data untuk Organizations.

Izin diperlukan

Untuk melihat informasi yang terakhir diakses di AWS Management Console, Anda harus memiliki kebijakan yang memberikan izin yang diperlukan.

Izin untuk informasi IAM

Untuk menggunakan IAM konsol untuk melihat informasi yang terakhir diakses untuk IAM pengguna, peran, atau kebijakan, Anda harus memiliki kebijakan yang mencakup tindakan berikut:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:Get*`
- `iam:List*`

Izin ini memungkinkan pengguna melihat hal berikut:

- Pengguna, kelompok, atau peran yang dilampirkan pada kebijakan terkelola
- Layanan yang dapat diakses pengguna atau peran
- Terakhir kali mereka mengakses layanan

- Terakhir kali mereka mencoba menggunakan tindakan Amazon,, Lambda EC2IAM, atau Amazon S3 tertentu

Untuk menggunakan AWS CLI atau AWS API untuk melihat informasi yang terakhir diaksesIAM, Anda harus memiliki izin yang cocok dengan operasi yang ingin Anda gunakan:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam:ListPoliciesGrantingServiceAccess`

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan melihat informasi yang IAM terakhir diakses. Selain itu, ini memungkinkan akses hanya-baca ke semua. IAM Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateServiceLastAccessedDetails",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
```

Izin untuk informasi AWS Organizations

Untuk menggunakan IAM konsol untuk melihat laporan untuk entitas root, OU, atau akun di Organizations, Anda harus memiliki kebijakan yang mencakup tindakan berikut:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:DescribeOrganizationalUnit`

- organizations:DescribePolicy
- organizations:ListChildren
- organizations:ListParents
- organizations:ListPoliciesForTarget
- organizations:ListRoots
- organizations:ListTargetsForPolicy

Untuk menggunakan AWS CLI atau AWS API untuk melihat layanan informasi yang terakhir diakses untuk Organizations, Anda harus memiliki kebijakan yang mencakup tindakan berikut:

- iam:GenerateOrganizationsAccessReport
- iam:GetOrganizationsAccessReport
- organizations:DescribePolicy
- organizations:ListChildren
- organizations:ListParents
- organizations:ListPoliciesForTarget
- organizations:ListRoots
- organizations:ListTargetsForPolicy

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan melihat layanan informasi yang terakhir diakses untuk Organizations. Selain itu, ini memungkinkan akses baca-saja ke seluruh Organisasi. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
```

```
    }
}
```

Anda juga dapat menggunakan kunci OrganizationsPolicyId kondisi iam: untuk memungkinkan pembuatan laporan hanya untuk kebijakan Organizations tertentu. Untuk contoh kebijakan, lihat IAM: Melihat informasi layanan yang terakhir diakses untuk kebijakan Organizations.

Memecahkan masalah aktivitas untuk dan entitas IAM Organizations

Dalam beberapa kasus, tabel informasi AWS Management Console terakhir Anda yang diakses mungkin kosong. Atau mungkin AWS API permintaan AWS CLI atau Anda mengembalikan sekumpulan informasi kosong atau bidang nol. Dalam kejadian ini, tinjau masalah berikut:

- Untuk informasi tindakan yang terakhir kali diakses, tindakan yang Anda harapkan untuk dilihat mungkin tidak akan dikembalikan dalam daftar. Ini dapat terjadi baik karena IAM identitas tidak memiliki izin untuk tindakan, atau AWS belum melacak tindakan untuk informasi yang terakhir diakses.
- Untuk IAM pengguna, pastikan bahwa pengguna memiliki setidaknya satu kebijakan inline atau terkelola yang dilampirkan, baik secara langsung atau melalui keanggotaan grup.
- Untuk IAM grup, verifikasi bahwa grup memiliki setidaknya satu kebijakan inline atau terkelola yang dilampirkan.
- Untuk IAM grup, laporan hanya menampilkan layanan informasi yang terakhir diakses untuk anggota yang menggunakan kebijakan grup untuk mengakses layanan. Untuk mempelajari apakah anggota menggunakan kebijakan lain, tinjau informasi yang terakhir diakses untuk pengguna tersebut.
- Untuk IAM peran, verifikasi bahwa peran tersebut memiliki setidaknya satu kebijakan sebaris atau terkelola yang dilampirkan.
- Untuk IAM entitas (pengguna atau peran), tinjau jenis kebijakan lain yang mungkin memengaruhi izin entitas tersebut. Ini termasuk kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas IAM izin, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Mengevaluasi kebijakan dalam satu akun.
- Untuk IAM kebijakan, pastikan bahwa kebijakan terkelola yang ditentukan dilampirkan ke setidaknya satu pengguna, grup dengan anggota, atau peran.
- Untuk entitas Organisasi (root, OU, atau akun), pastikan bahwa Anda masuk menggunakan kredensial akun manajemen Organisasi.
- Verifikasi SCPsyang diaktifkan di root organisasi Anda.

- Informasi tindakan yang terakhir diakses hanya tersedia untuk tindakan yang tercantum dalamIAMtindakan terakhir diakses layanan informasi dan tindakan.

Saat Anda membuat perubahan, tunggu setidaknya empat jam hingga aktivitas muncul di laporan IAM konsol Anda. Jika Anda menggunakan AWS CLI atau AWS API, Anda harus membuat laporan baru untuk melihat informasi yang diperbarui.

Tempat AWS melacak informasi terakhir yang diakses

AWS mengumpulkan informasi yang terakhir diakses untuk AWS Wilayah standar. Saat AWS menambahkan Wilayah tambahan, Wilayah tersebut ditambahkan ke tabel berikut, termasuk tanggal yang AWS mulai melacak informasi di setiap Wilayah.

- Informasi layanan — Periode pelacakan untuk layanan setidaknya 400 hari, atau kurang jika Wilayah Anda mulai melacak fitur ini dalam 400 hari terakhir.
- Informasi tindakan – Periode pelacakan untuk tindakan manajemen Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk Amazon EC2IAM,, dan tindakan manajemen Lambda dimulai pada 7 April 2021. Periode pelacakan untuk tindakan manajemen semua layanan lainnya dimulai pada 23 Mei 2023. Jika tanggal pelacakan suatu Wilayah lebih lambat dari 23 Mei 2023, maka tindakan yang terakhir diakses informasi dari Wilayah tersebut akan dimulai di kemudian hari.

Nama Wilayah	Region	Tanggal mulai pelacakan
AS Timur (Ohio)	as-timur-2	27 Oktober 2017
AS Timur (Virginia Utara)	as-timur-1	1 Oktober 2015
AS Barat (California Utara)	as-barat-1	1 Oktober 2015
AS Barat (Oregon)	as-barat-2	1 Oktober 2015
Afrika (Cape Town)	af-selatan-1	22 April 2020
Asia Pasifik (Hong Kong)	ap-timur-1	24 April 2019
Asia Pasifik (Hyderabad)	ap-south-2	22 November 2022
Asia Pasifik (Jakarta)	ap-southeast-3	13 Desember 2021

Nama Wilayah	Region	Tanggal mulai pelacakan
Asia Pasifik (Melbourne)	ap-southeast-4	23 Januari 2023
Asia Pasifik (Mumbai)	ap-selatan-1	27 Juni 2016
Asia Pacific (Osaka)	ap-northeast-3	Februari 11, 2018
Asia Pasifik (Seoul)	ap-timur laut-2	6 Januari 2016
Asia Pasifik (Singapura)	ap-tenggara-1	1 Oktober 2015
Asia Pasifik (Sydney)	ap-tenggara-2	1 Oktober 2015
Asia Pasifik (Tokyo)	ap-timur laut-1	1 Oktober 2015
Kanada (Pusat)	ca-sentral-1	28 Oktober 2017
Eropa (Frankfurt)	eu-sentral-1	1 Oktober 2015
Eropa (Irlandia)	eu-barat-1	1 Oktober 2015
Eropa (London)	eu-barat-2	28 Oktober 2017
Eropa (Milan)	eu-selatan-1	28 April 2020
Eropa (Paris)	eu-barat-3	18 Desember 2017
Eropa (Spanyol)	eu-south-2	15 November 2022
Eropa (Stockholm)	eu-utara-1	12 Desember 2018
Eropa (Zürich)	eu-central-2	8 November 2022
Israel (Tel Aviv)	il-central-1	1 Agustus 2023
Timur Tengah (Bahrain)	me-selatan-1	29 Juli 2019
Timur Tengah (UAE)	me-central-1	30 Agustus 2022
Amerika Selatan (Sao Paulo)	sa-timur-1	11 Desember 2015

Nama Wilayah	Region	Tanggal mulai pelacakan
AWS GovCloud (AS-Timur)	us-gov-east-1	Juli 1, 2023
AWS GovCloud (AS-Barat)	us-gov-west-1	Juli 1, 2023

Jika region tidak tercantum dalam tabel sebelumnya, maka region tersebut belum memberikan informasi yang terakhir diakses.

AWS Wilayah adalah kumpulan AWS sumber daya di wilayah geografis. Region-region dikelompokkan menjadi beberapa bagian. Region standar adalah region yang termasuk ke bagian `aws`. Untuk informasi selengkapnya tentang partisi yang berbeda, lihat Format Amazon Resource Names (ARNs) di. Referensi Umum AWS Untuk informasi selengkapnya tentang Wilayah, lihat Tentang AWS Wilayah juga di Referensi Umum AWS.

Lihat informasi yang terakhir diakses untuk IAM

Anda dapat melihat informasi yang terakhir diakses untuk IAM menggunakan AWS Management Console, AWS CLI, atau AWS API. Lihat daftar layanan dan tindakan mereka yang informasi terakhir diakses ditampilkan. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Anda dapat melihat informasi untuk jenis sumber daya berikut diIAM. Dalam setiap kejadian, informasi tersebut mencakup layanan yang diizinkan untuk periode pelaporan tertentu:

- Pengguna — Lihat terakhir kali pengguna mencoba mengakses setiap layanan yang diizinkan.
- Grup pengguna — Melihat informasi tentang terakhir kali anggota grup pengguna mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total anggota yang mencoba mengaksesnya.
- Peran – Lihat terakhir kali seseorang menggunakan peran tersebut dalam upaya mengakses setiap layanan yang diizinkan.
- Kebijakan – Melihat informasi terakhir kali pengguna atau peran mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total entitas yang mencoba mengaksesnya.

> **Note**
>
> Sebelum Anda melihat informasi akses untuk sumber dayaIAM, pastikan Anda memahami periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi untuk informasi Anda. Untuk detail selengkapnya, lihat the section called “Hal yang perlu diketahui tentang informasi yang terakhir diakses”.

Melihat informasi untuk IAM (konsol)

Anda dapat melihat informasi yang terakhir diakses IAM di tab Access Advisor di IAM konsol.

Melihat informasi untuk IAM (konsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, Peran, atau Kebijakan.
3. Pilih pengguna, grup pengguna, peran, atau nama kebijakan untuk halaman Ringkasan dan pilih tab Penasihat Akses. Lihat informasi berikut, berdasarkan sumber daya yang Anda pilih:

   - Grup pengguna - Lihat daftar layanan yang dapat diakses oleh anggota grup pengguna. Anda juga dapat melihat kapan anggota terakhir mengakses layanan, kebijakan grup pengguna apa yang mereka gunakan, dan anggota grup pengguna mana yang membuat permintaan. Pilih nama kebijakan untuk dipelajari baik merupakan kebijakan terkelola atau kebijakan grup pengguna inline. Pilih nama anggota grup pengguna untuk melihat semua anggota grup pengguna dan kapan mereka terakhir kali mengakses layanan.
   - Pengguna – Lihat daftar layanan yang dapat diakses pengguna. Anda juga dapat melihat kapan mereka terakhir mengakses layanan, dan kebijakan apa yang saat ini terkait dengan pengguna. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola, kebijakan pengguna inline, atau kebijakan inline untuk grup pengguna.
   - Peran – Lihat daftar layanan yang dapat diakses oleh peran, kapan peran terakhir mengakses layanan, dan kebijakan apa yang digunakan. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola atau kebijakan peran selaras.
   - Kebijakan – Lihat daftar layanan dengan tindakan yang diperbolehkan dalam kebijakan. Anda juga dapat melihat kapan kebijakan terakhir digunakan untuk mengakses layanan, dan entitas (pengguna atau peran) mana yang menggunakan kebijakan tersebut. Tanggal Terakhir diakses juga mencakup kapan akses diberikan ke kebijakan ini melalui kebijakan lain. Pilih

nama entitas untuk dipelajari yang sudah terlampirkan dengan kebijakan ini dan kapan terakhir kali mereka mengakses layanan.

4. Di kolom Layanan pada tabel, pilih nama salah satu layanan yang menyertakan informasi tindakan yang terakhir diakses untuk melihat daftar tindakan manajemen yang coba diakses IAM entitas. Anda dapat melihat Wilayah AWS dan stempel waktu yang menunjukkan kapan seseorang terakhir mencoba melakukan tindakan.
5. Kolom Terakhir diakses ditampilkan untuk layanan dan tindakan manajemen layanan yang menyertakan informasi tindakan yang terakhir diakses. Tinjau hasil-hasil berikut yang kemungkinan muncul dalam kolom ini. Hasil ini bervariasi tergantung pada apakah layanan atau tindakan diizinkan, diakses, dan apakah itu dilacak oleh AWS untuk informasi yang terakhir diakses.

   <jumlah> hari yang lalu

   Jumlah hari sejak layanan atau tindakan digunakan dalam periode pelacakan. Periode pelacakan layanan adalah selama 400 hari terakhir. Periode pelacakan untuk tindakan Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk Amazon EC2IAM,, dan tindakan Lambda dimulai pada 7 April 2021. Periode pelacakan untuk semua layanan lainnya dimulai pada 23 Mei 2023. Untuk mempelajari selengkapnya tentang melacak tanggal mulai untuk masing-masing Wilayah AWS, lihatTempat AWS melacak informasi terakhir yang diakses.

   Tidak diakses dalam periode pelacakan

   Layanan atau tindakan yang dilacak belum digunakan oleh entitas dalam periode pelacakan.

   Anda dapat memiliki izin untuk tindakan yang tidak muncul dalam daftar. Ini dapat terjadi jika informasi pelacakan untuk tindakan saat ini tidak disertakan oleh AWS. Anda tidak diperkenankan mengambil keputusan izin hanya berdasarkan ketiadaan informasi pelacakan. Alih-alih, kami menyarankan agar Anda menggunakan informasi ini untuk menginformasikan dan mendukung strategi Anda secara keseluruhan untuk memberikan privilese paling sedikit. Periksa kebijakan Anda untuk mengonfirmasi bahwa tingkat akses sesuai.

Melihat informasi untuk IAM(AWS CLI)

Anda dapat menggunakan AWS CLI untuk mengambil informasi tentang terakhir kali IAM sumber daya digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, EC2

AmazonIAM, dan Lambda. IAMSumber daya dapat berupa pengguna, grup pengguna, peran, atau kebijakan.

Untuk melihat informasi IAM(AWS CLI)

1. Buat laporan. Permintaan harus menyertakan IAM sumber daya (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. ARN Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat detail akses baik untuk layanan maupun layanan beserta tindakan. Permintaan tersebut menghasilkan `job-id` yang kemudian dapat Anda gunakan di operasi `get-service-last-accessed-details` dan `get-service-last-accessed-details-with-entities` untuk memonitor `job-status` hingga pekerjaan selesai.

   - aws iam generate-service-last -detail yang dapat diakses
2. Dapatkan detail laporan menggunakan parameter `job-id` dari langkah sebelumnya.

   - aws iam get-service-last -detail yang dapat diakses

   Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `generate-service-last-accessed-details` operasi:

   - Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir pengguna dan pengguna. ARN
   - Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ini juga mengembalikan ARN pengguna itu dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan. Gunakan GetServiceLastAccessedDetailsWithEntitiesoperasi untuk mengambil daftar semua anggota.
   - Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir peran dan peran. ARN
   - Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna atau peran) untuk mengakses layanan menggunakan kebijakan. Ini juga mengembalikan ARN entitas itu dan jumlah total entitas yang mencoba mengakses.

3. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini mengembalikan daftar entitas dengan masing-masing entitasARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - aws iam get-service-last - accessed-details-with-entities

4. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas IAM izin, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Mengevaluasi kebijakan dalam satu akun.

   - aws iam list-policies-granting -layanan-akses

Melihat informasi untuk IAM (AWS API)

Anda dapat menggunakan AWS API untuk mengambil informasi tentang terakhir kali IAM sumber daya digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, EC2 AmazonIAM, dan Lambda. IAMSumber daya dapat berupa pengguna, grup pengguna, peran, atau kebijakan. Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat baik layanan maupun layanan beserta tindakan.

Untuk melihat informasi untuk IAM (AWS API)

1. Buat laporan. Permintaan harus menyertakan IAM sumber daya (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. ARN Ia menghasilkan `JobId` yang kemudian dapat Anda gunakan di operasi `GetServiceLastAccessedDetails` dan `GetServiceLastAccessedDetailsWithEntities` untuk memonitor `JobStatus` hingga pekerjaan selesai.

   - GenerateServiceLastAccessedDetails

2. Dapatkan detail laporan menggunakan parameter `JobId` dari langkah sebelumnya.

   - GetServiceLastAccessedDetails

Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `GenerateServiceLastAccessedDetails` operasi:

- Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir pengguna dan pengguna. ARN
- Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ini juga mengembalikan ARN pengguna itu dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan. Gunakan GetServiceLastAccessedDetailsWithEntitiesoperasi untuk mengambil daftar semua anggota.
- Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi mengembalikan tanggal dan waktu upaya terakhir peran dan peran. ARN
- Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna atau peran) untuk mengakses layanan menggunakan kebijakan. Ini juga mengembalikan ARN entitas itu dan jumlah total entitas yang mencoba mengakses.

3. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini mengembalikan daftar entitas dengan masing-masing entitasARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - GetServiceLastAccessedDetailsWithEntities

4. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas IAM izin, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Mengevaluasi kebijakan dalam satu akun.

   - ListPoliciesGrantingServiceAccess

Lihat informasi yang terakhir diakses untuk Organizations

Anda dapat melihat layanan informasi yang terakhir diakses untuk AWS Organizations menggunakan IAM konsol, AWS CLI, atau AWS API. Untuk informasi penting tentang data, izin yan diperlukan, pemecahan masalah, dan region yang didukung lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Saat masuk ke IAM konsol menggunakan kredenal akun AWS Organizations manajemen, Anda dapat melihat informasi untuk entitas apa pun di organisasi Anda. Organizations Entities mencakup root organisasi, unit organisasi (OUs), dan akun. Anda juga dapat menggunakan IAM konsol untuk melihat informasi untuk kebijakan kontrol layanan apa pun (SCPs) di organisasi Anda. IAMmenampilkan daftar layanan yang diizinkan oleh siapa pun SCPs yang berlaku untuk entitas. Untuk setiap layanan, Anda dapat melihat informasi aktivitas akun terbaru untuk entitas organisasi terpilih atau anak-anak entitas.

Saat Anda menggunakan AWS CLI atau AWS API dengan kredensi akun manajemen, Anda dapat membuat laporan untuk entitas atau kebijakan apa pun di organisasi Anda. Laporan terprogram untuk entitas mencakup daftar layanan yang diizinkan oleh siapa pun SCPs yang berlaku untuk entitas. Untuk setiap layanannya, laporan ini mencakup aktivitas terbaru akun di entitas Organisasi yang spesifik atau cabang dari entitas tersebut.

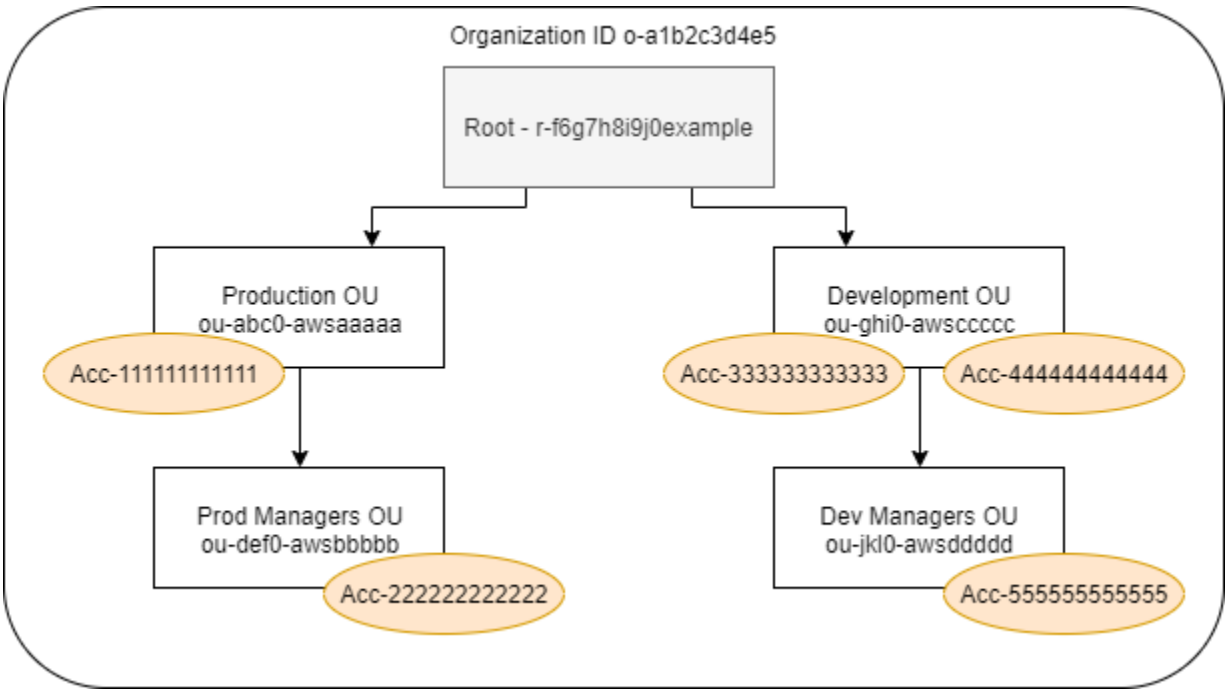
Saat Anda membuat laporan program untuk suatu kebijakan, Anda harus menetapkan entitas Organisasi. Laporan ini mencakup daftar layanan yang diizinkan oleh yang ditentukanSCP. Untuk setiap layanan, ia mencakup aktivitas akun terbaru dalam entitas atau anak entitas yang diberikan izin oleh kebijakan tersebut. Untuk informasi selengkapnya, lihat aws iam generate-organizations-access -report atau. GenerateOrganizationsAccessReport

Sebelum melihat laporan, pastikan bahwa Anda memahami persyaratan dan informasi akun manajemen, periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi. Untuk lebih detailnya, lihat the section called “Hal yang perlu diketahui tentang informasi yang terakhir diakses”.

Memahami jalur entitas AWS Organizations

Saat Anda menggunakan AWS CLI atau AWS API untuk membuat laporan AWS Organizations akses, Anda harus menentukan jalur entitas. Jalur adalah representasi teks dari struktur entitas Organisasi.

Anda dapat membangun jalur entitas menggunakan struktur Organisasi Anda yang sudah Anda pahami. Misalnya, asumsikan bahwa Anda memiliki struktur organisasi berikut AWS Organizations.



Jalur untuk Manajer Dev OU dibangun menggunakan organisasi, root, dan semua OUs di jalur ke dan termasuk OU. IDs

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsddddd/
```

Jalur untuk akun di OU Produksi dibangun menggunakan organisasi, root, OU, dan nomor akun. IDs

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> Organisasi IDs secara global unik tetapi OU IDs dan root IDs hanya unik dalam suatu organisasi. Ini berarti bahwa tidak ada dua organisasi yang memiliki ID organisasi yang sama. Namun, organisasi lain mungkin memiliki OU atau root dengan ID yang sama seperti milik Anda. Kami merekomendasikan agar Anda selalu menyertakan ID organisasi saat menentukan OU atau root.

Melihat informasi untuk Organisasi (konsol)

Anda dapat menggunakan IAM konsol untuk melihat layanan informasi yang terakhir diakses untuk root, OU, akun, atau kebijakan Anda.

Untuk melihat informasi root (konsol)

1. Masuk ke kredensial akun manajemen AWS Management Console menggunakan Organizations, dan buka IAM konsol di. https://console.aws.amazon.com/iam/
2. Di panel navigasi di bawah bagian Mengakses laporan, pilih Aktivitas organisasi.
3. Di halaman Aktivitas organisasi, pilih Akar.
4. Di tab Detail dan aktivitas, lihat bagian Laporan akses layanan. Informasi tersebut mencakup daftar layanan yang diizinkan oleh kebijakan yang dilampirkan secara langsung pada akar. Informasi tersebut menunjukkan kepada Anda akun mana yang terakhir diakses dan kapan. Untuk detail selengkapnya tentang prinsipal mana yang mengakses layanan, masuk sebagai administrator di akun tersebut dan lihat informasi IAM layanan yang terakhir diakses.
5. Pilih SCPs tab Terlampir untuk melihat daftar kebijakan kontrol layanan (SCPs) yang dilampirkan ke root. IAMmenunjukkan kepada Anda jumlah entitas target yang dilampirkan setiap kebijakan. Anda dapat menggunakan informasi ini untuk memutuskan mana yang akan SCP ditinjau.
6. Pilih nama SCP untuk melihat semua layanan yang diizinkan oleh kebijakan tersebut. Untuk setiap layanan, lihat dari akun mana layanan tersebut terakhir diakses, dan kapan.
7. Pilih Edit AWS Organizations untuk melihat detail tambahan dan mengedit SCP di konsol Organizations. Untuk informasi selengkapnya, lihat Memperbarui SCP di Panduan AWS Organizations Pengguna.

Untuk melihat informasi untuk OU atau akun (konsol)

1. Masuk ke kredensial akun manajemen AWS Management Console menggunakan Organizations, dan buka IAM konsol di. https://console.aws.amazon.com/iam/
2. Di panel navigasi di bawah bagian Mengakses laporan, pilih Aktivitas organisasi.
3. Di Aktivitas organisasi, buka lebih jauh ke struktur organisasi Anda. Kemudian pilih nama OU atau akun apa pun yang ingin Anda lihat kecuali akun manajemen.
4. Di tab Detail dan aktivitas, lihat bagian Laporan akses layanan. Informasi tersebut mencakup daftar layanan yang diizinkan oleh SCPs terlampir pada OU atau akun dan semua orang tuanya. Informasi tersebut menunjukkan kepada Anda akun mana yang terakhir diakses dan kapan. Untuk detail selengkapnya tentang prinsipal mana yang mengakses layanan, masuk sebagai administrator di akun tersebut dan lihat informasi IAM layanan yang terakhir diakses.
5. Pilih SCPs tab Terlampir untuk melihat daftar kebijakan kontrol layanan (SCPs) yang dilampirkan langsung ke OU atau akun. IAMmenunjukkan kepada Anda jumlah entitas target yang

dilampirkan setiap kebijakan. Anda dapat menggunakan informasi ini untuk memutuskan mana yang akan SCP ditinjau.

6. Pilih nama SCP untuk melihat semua layanan yang diizinkan oleh kebijakan tersebut. Untuk setiap layanan, lihat dari akun mana layanan tersebut terakhir diakses, dan kapan.
7. Pilih Edit AWS Organizations untuk melihat detail tambahan dan mengedit SCP di konsol Organizations. Untuk informasi selengkapnya, lihat Memperbarui SCP di Panduan AWS Organizations Pengguna.

Untuk melihat informasi akun manajemen (konsol)

1. Masuk ke kredensial akun manajemen AWS Management Console menggunakan Organizations, dan buka IAM konsol di. https://console.aws.amazon.com/iam/
2. Di panel navigasi di bawah bagian Mengakses laporan, pilih Aktivitas organisasi.
3. Di halaman Aktivitas organisasi buka lebih lanjut struktur organisasi Anda dan pilih nama akun manajemen Anda.
4. Di tab Detail dan aktivitas, lihat bagian Laporan akses layanan. Informasi ini mencakup daftar semua layanan AWS . Akun manajemen tidak dibatasi olehSCPs. Informasi tersebut menunjukkan kepada Anda apakah akun terakhir mengakses layanan dan waktunya. Untuk detail selengkapnya tentang prinsipal mana yang mengakses layanan, masuk sebagai administrator di akun tersebut dan lihat informasi IAM layanan yang terakhir diakses.
5. Pilih SCPs tab Terlampir untuk mengonfirmasi bahwa tidak ada yang terlampir SCPs karena akun tersebut adalah akun manajemen.

Untuk melihat informasi untuk kebijakan (konsol)

1. Masuk ke kredensial akun manajemen AWS Management Console menggunakan Organizations, dan buka IAM konsol di. https://console.aws.amazon.com/iam/
2. Di panel navigasi di bawah bagian Laporan akses, pilih Kebijakan kontrol layanan (SCPs).
3. Pada halaman Kebijakan kontrol layanan (SCPs), lihat daftar kebijakan di organisasi Anda. IAM menunjukkan jumlah entitas target yang terlampirkan dengan masing-masing kebijakan.
4. Pilih nama SCP untuk melihat semua layanan yang diizinkan oleh kebijakan tersebut. Untuk setiap layanan, lihat dari akun mana layanan tersebut terakhir diakses, dan kapan.

5. Pilih Edit AWS Organizations untuk melihat detail tambahan dan mengedit SCP di konsol Organizations. Untuk informasi selengkapnya, lihat Memperbarui SCP di Panduan AWS Organizations Pengguna.

Melihat informasi untuk Organizations (AWS CLI)

Anda dapat menggunakan AWS CLI untuk mengambil layanan informasi yang terakhir diakses untuk root, OU, akun, atau kebijakan Organisasi Anda.

Untuk melihat layanan Organizations informasi terakhir diakses (AWS CLI)

1. Gunakan kredensi akun manajemen Organisasi Anda dengan izin yang diperlukan dan IAM Organizations, dan konfirmasikan bahwa itu SCPs diaktifkan untuk root Anda. Untuk informasi selengkapnya, lihat Hal yang perlu diketahui tentang informasi yang terakhir diakses.
2. Buat laporan. Permintaan harus menyertakan alur entitas Organisasi (root, OU, atau akun) yang Anda inginkan laporannya. Anda secara opsional dapat menyertakan parameter `organization-policy-id` untuk melihat laporan kebijakan tertentu. Perintah menyatakan `job-id` yang kemudian dapat Anda gunakan di `get-organizations-access-report` perintah untuk memonitor `job-status` hingga pekerjaan selesai.

   - aws iam generate-organizations-access -laporan
3. Dapatkan detail laporan menggunakan parameter `job-id` dari langkah sebelumnya.

   - aws iam get-organizations-access -laporan

   Perintah ini menghasilkan daftar layanan yang dapat diakses oleh anggota entitas. Untuk setiap layanan, perintah menyatakan tanggal dan waktu upaya terakhir anggota akun dan alur entitas akun. Ia juga menyatakan jumlah total layanan yang tersedia untuk diakses dan jumlah layanan yang tidak diakses. Jika Anda menentukan pilihan `organizations-policy-id` parameter, maka layanan yang tersedia untuk diakses adalah layanan yang diizinkan oleh kebijakan tertentu.

Melihat informasi untuk Organizations (AWS API)

Anda dapat menggunakan AWS API untuk mengambil layanan informasi yang terakhir diakses untuk root, OU, akun, atau kebijakan Organisasi Anda.

Untuk melihat layanan Organizations informasi terakhir diakses (AWS API)

1. Gunakan kredensi akun manajemen Organisasi Anda dengan izin yang diperlukan dan IAM Organizations, dan konfirmasikan bahwa itu SCPs diaktifkan untuk root Anda. Untuk informasi selengkapnya, lihat Hal yang perlu diketahui tentang informasi yang terakhir diakses.
2. Buat laporan. Permintaan harus menyertakan alur entitas Organisasi (root, OU, atau akun) yang Anda inginkan laporannya. Anda secara opsional dapat menyertakan parameter `OrganizationsPolicyId` untuk melihat laporan kebijakan tertentu. Operasi menghasilkan `JobId` yang kemudian dapat Anda gunakan di `GetOrganizationsAccessReport` operasi untuk memonitor `JobStatus` hingga pekerjaan selesai.

   - GenerateOrganizationsAccessReport
3. Dapatkan detail laporan menggunakan parameter `JobId` dari langkah sebelumnya.

   - GetOrganizationsAccessReport

   Operasi ini menyatakan daftar layanan yang dapat diakses oleh anggota entitas. Untuk setiap layanan, perintah menyatakan tanggal dan waktu upaya terakhir anggota akun dan alur entitas akun. Ia juga menyatakan jumlah total layanan yang tersedia untuk diakses, dan jumlah layanan yang tidak diakses. Jika Anda menentukan pilihan `OrganizationsPolicyId` parameter, maka layanan yang tersedia untuk diakses adalah layanan yang diizinkan oleh kebijakan tertentu.

Contoh alur perencanaan untuk menggunakan informasi yang terakhir diakses

Anda dapat menggunakan informasi yang diakses terakhir untuk membuat keputusan tentang izin yang Anda berikan kepada IAM entitas atau AWS Organizations entitas Anda. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

> **Note**
>
> Sebelum Anda melihat informasi akses untuk entitas atau kebijakan di IAM atau AWS Organizations, pastikan Anda memahami periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi untuk data Anda. Untuk lebih detailnya, lihat the section called “Hal yang perlu diketahui tentang informasi yang terakhir diakses”.

Sebagai administrator Anda bebas untuk menyeimbangkan ketersediaan akses dan hak istimewa terkecil untuk perusahaan Anda.

Menggunakan informasi untuk mengurangi izin untuk grup IAM

Anda dapat menggunakan informasi yang diakses terakhir untuk mengurangi izin IAM grup agar hanya menyertakan layanan yang dibutuhkan pengguna Anda. Metode ini merupakan langkah penting dalam memberikan hak istimewa terkecil pada tingkat layanan.

Sebagai contoh, Paulo Santos adalah administrator yang bertanggung jawab untuk mendefinisikan izin AWS pengguna untuk Contoh Corp. Perusahaan ini baru saja mulai menggunakan AWS, dan tim pengembangan perangkat lunak belum menentukan layanan apa yang AWS akan mereka gunakan. Paulo ingin memberikan izin kepada tim untuk hanya mengakses layanan yang mereka butuhkan, tetapi karena itu belum ditentukan, dia sementara memberikan izin penggunaan daya. Kemudian dia menggunakan informasi yang diakses terakhir untuk mengurangi izin kelompok.

Paulo membuat kebijakan terkelola bernama `ExampleDevelopment` menggunakan JSON teks berikut. Lalu ia melampirkannya ke kelompok bernama `Development` dan menambahkan semua developer ke kelompok.

> **Note**
>
> Pengguna daya Paulo mungkin membutuhkan izin `iam:CreateServiceLinkedRole` untuk menggunakan beberapa layanan dan fitur. Dia mengerti bahwa dia menambahkan izin ini memungkinkan pengguna membuat peran terkait layanan apa pun. Dia menerima risiko ini untuk pengguna dayanya.

```
{

    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
```

```
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Paulo memutuskan untuk menunggu selama 90 hari sebelum melihat informasi yang terakhir diakses untuk `Development` menggunakan AWS Management Console. Dia melihat daftar layanan yang diakses anggota grup. Dia mengetahui bahwa pengguna mengakses lima layanan dalam seminggu terakhir: AWS CloudTrail, Amazon CloudWatch Logs, Amazon, EC2 AWS KMS, dan Amazon S3. Mereka mengakses beberapa layanan lain ketika mereka pertama kali mengevaluasi AWS, tetapi tidak sejak saat itu.

Paulo memutuskan untuk mengurangi izin kebijakan untuk memasukkan hanya lima layanan tersebut dan tindakan yang diperlukan dan IAM Organizations. Dia mengedit `ExampleDevelopment` kebijakan menggunakan JSON teks berikut.

> **Note**
>
> Pengguna daya Paulo mungkin membutuhkan izin `iam:CreateServiceLinkedRole` untuk menggunakan beberapa layanan dan fitur. Dia mengerti bahwa dia menambahkan izin ini memungkinkan pengguna membuat peran terkait layanan apa pun. Dia menerima risiko ini untuk pengguna dayanya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk lebih mengurangi izin, Paulo dapat melihat kegiatan akun di AWS CloudTrail Riwayat peristiwa. Di sana, ia dapat melihat informasi kegiatan terperinci yang dapat ia gunakan untuk mengurangi izin kebijakan untuk hanya mencakup tindakan dan sumber daya yang dibutuhkan oleh developer. Untuk informasi selengkapnya, lihat Melihat CloudTrail Acara di CloudTrail Konsol di Panduan AWS CloudTrail Pengguna.

Menggunakan informasi untuk mengurangi izin bagi pengguna IAM

Anda dapat menggunakan informasi yang diakses terakhir untuk mengurangi izin bagi IAM pengguna individu.

Misalnya, Martha Rivera adalah administrator TI yang bertanggung jawab untuk memastikan bahwa orang-orang di perusahaannya tidak memiliki izin berlebih AWS . Sebagai bagian dari pemeriksaan keamanan berkala, ia meninjau izin semua IAM pengguna. Salah satu pengguna ini adalah developer aplikasi bernama Nikhil Jayashankar, yang sebelumnya telah memegang peran sebagai teknisi IT Karena perubahan dalam persyaratan pekerjaan, Nikhil adalah anggota baik grup `app-dev` maupun grup `security-team`. `app-dev`Grup untuk pekerjaan barunya memberikan izin ke beberapa layanan termasuk Amazon, Amazon, Auto ScalingEBS, EC2 Amazon S3, Route 53, dan Elastic Transcoder. `security-team`Kelompok untuk pekerjaan lamanya memberikan izin untuk IAM dan. CloudTrail

Sebagai administrator, Martha masuk ke IAM konsol dan memilih Pengguna, memilih nama**nikhilj**, dan kemudian memilih tab Terakhir Diakses.

Martha meninjau kolom Terakhir Diakses dan pemberitahuan bahwa Nikhil belum baru-baru ini mengaksesIAM,, Route 53 CloudTrail, Amazon Elastic Transcoder, dan sejumlah layanan lainnya. AWS Nikhil telah mengakses Amazon S3. Martha memilih S3 dari daftar layanan dan mengetahui bahwa Nikhil telah melakukan beberapa tindakan Amazon S3 dalam dua minggu terakhir. `List` Di dalam perusahaannya, Martha menegaskan bahwa Nikhil tidak memiliki kebutuhan bisnis untuk mengakses IAM dan CloudTrail lagi karena dia tidak lagi menjadi anggota tim keamanan internal.

Martha sekarang siap bertindak atas layanan tersebut dan tindakan informasi yang terakhir diakses. Namun, tidak seperti grup pada contoh sebelumnya, IAM pengguna seperti `nikhilj` mungkin tunduk pada beberapa kebijakan dan menjadi anggota dari beberapa grup. Martha harus berhati-hati agar tidak mengganggu akses `nikhilj` atau anggota grup lainnya. Sebagai tambahan dalam hal untuk mempelajari apa akses yang harus dimiliki Nikhil, ia harus menentukan cara dia menerima izin ini.

Martha memilih tab Izin, tempatnya melihat kebijakan mana yang terkait secara langsung ke `nikhilj` dan yang terlampir di grup. Ia membuka lebih lanjut setiap kebijakan dan melihat ringkasan kebijakan untuk mempelajari kebijakan yang memungkinkan akses ke layanan yang tidak digunakan oleh Nikhil:

- IAM— Kebijakan yang `IAMFullAccess` AWS dikelola dilampirkan langsung ke `nikhilj` dan dilampirkan ke `security-team` grup.
- CloudTrail — Kebijakan `AWS CloudTrailReadOnlyAccess` AWS terkelola dilampirkan pada `security-team` grup.
- Route 53 – Kebijakan terkelola pelanggan `App-Dev-Route53` dilampirkan pada grup `app-dev`.
- Transkoder Elastis – Kebijakan terkelola pelanggan `App-Dev-ElasticTranscoder` terlampir pada grup `app-dev`.

Martha memutuskan untuk menghapus kebijakan `IAMFullAccess` AWS terkelola yang dilampirkan `nikhilj` langsung. Dia juga menghapus keanggotaan Nikhil ke grup `security-team`. Kedua tindakan ini menghapus akses yang tidak perlu ke IAM dan CloudTrail.

Izin Nikhil untuk mengakses Route 53 dan Transkoder Elastis diberikan oleh grup `app-dev`. Meskipun Nikhil tidak menggunakan layanan tersebut, anggota lain kelompok tersebut mungkin saja demikian. Martha meninjau informasi yang terakhir diakses untuk grup `app-dev` dan mempelajari

bahwa beberapa anggota mengakses Route 53 dan Amazon S3 baru-baru ini. Namun, tidak ada anggota kelompok yang telah mengakses Transkoder Elastis tahun lalu. Dia menghapus kebijakan pengelolaan pelanggan `App-Dev-ElasticTranscoder` dari grup.

Martha kemudian meninjau informasi yang terakhir diakses untuk `App-Dev-ElasticTranscoder` kebijakan terkelola pelanggan. Dia belajar bahwa kebijakan itu tidak melekat pada IAM identitas lain. Ia menginvestigasi dalam perusahaannya untuk memastikan bahwa kebijakan tersebut tidak diperlukan di waktu yang akan datang, kemudian ia menghapusnya.

Menggunakan informasi sebelum menghapus sumber daya IAM

Anda dapat menggunakan informasi yang terakhir diakses sebelum menghapus IAM sumber daya untuk memastikan bahwa sejumlah waktu telah berlalu sejak seseorang terakhir menggunakan sumber daya tersebut. Ini berlaku untuk pengguna, grup, peran, dan kebijakan. Untuk mempelajari lebih lanjut tentang tindakan ini, lihat topik berikut:

- Pengguna – Menghapus pengguna
- Grup – Menghapus grup
- Peran – Menghapus peran
- Kebijakan – Menghapus kebijakan terkelola (ini juga akan menghapus kebijakan dari identitas)

Menggunakan informasi sebelum menyunting kebijakan IAM

Anda dapat meninjau informasi yang terakhir diakses untuk IAM identitas (pengguna, grup, atau peran), atau untuk IAM kebijakan sebelum mengedit kebijakan yang memengaruhi sumber daya tersebut. Ini penting karena Anda tidak ingin menghapus akses untuk orang yang menggunakannya.

Misalnya, Arnav Desai adalah pengembang dan AWS administrator untuk Example Corp. Ketika timnya mulai menggunakan AWS, mereka memberi semua pengembang akses power-user yang memungkinkan mereka akses penuh ke semua layanan kecuali dan Organizations. IAM Sebagai langkah pertama untuk memberikan hak istimewa paling rendah, Arnav ingin menggunakan AWS CLI untuk meninjau kebijakan yang dikelola dalam akunnya.

Untuk melakukannya, Arnav terlebih dahulu mencantumkan kebijakan izin yang dikelola pelanggan di akunnya yang terlampir ke suatu identitas, menggunakan perintah berikut:

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

Dari tanggapan, ia menangkap ARN untuk setiap kebijakan. Arnav membuat laporan untuk informasi yang terakhir diakses bagi setiap kebijakan dengan menggunakan perintah berikut.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Dari respons tersebut, dia mengambil ID dari laporan yang dihasilkan dari bidang `JobId`. Arnav kemudian melakukan jajak pendapat perintah berikut sampai bidang `JobStatus` menghasilkan nilai `COMPLETED` atau `FAILED`. Jika tugas gagal, ia akan menangkap kesalahan.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Ketika pekerjaan memiliki status`COMPLETED`, Arnav mem-parsing isi array yang diformat. JSON `ServicesLastAccessed`

```
 "ServicesLastAccessed": [
        {
            "TotalAuthenticatedEntities": 1,
            "LastAuthenticated": 2018-11-01T21:24:33.222Z,
            "ServiceNamespace": "dynamodb",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
            "ServiceName": "Amazon DynamoDB"
        },

        {
            "TotalAuthenticatedEntities": 0,
            "ServiceNamespace": "ec2",
            "ServiceName": "Amazon EC2"
        },

        {
            "TotalAuthenticatedEntities": 3,
            "LastAuthenticated": 2018-08-25T15:29:51.156Z,
            "ServiceNamespace": "s3",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
            "ServiceName": "Amazon S3"
        }
    ]
```

Dari informasi ini, Arnav mengetahui bahwa `ExamplePolicy1` kebijakan tersebut memungkinkan akses ke tiga layanan, Amazon DynamoDB, Amazon S3, dan Amazon. EC2 IAMPengguna bernama

`IAMExampleUser` terakhir mencoba mengakses DynamoDB pada 1 November, dan seseorang menggunakan peran `IAMExampleRole` tersebut untuk mencoba mengakses Amazon S3 pada 25 Agustus. Ada pula dua entitas lagi yang mencoba mengakses Amazon S3 tahun lalu. Namun, tidak ada yang mencoba mengakses Amazon EC2 dalam setahun terakhir.

Ini berarti bahwa Arnav dapat dengan aman menghapus EC2 tindakan Amazon dari kebijakan. Arnav ingin meninjau JSON dokumen saat ini untuk kebijakan tersebut. Pertama, ia harus menentukan nomor versi kebijakan menggunakan perintah berikut.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Dari respons tersebut, Arnav mengumpulkan nomor versi default saat ini dari himpunan `Versions`. Dia kemudian menggunakan nomor versi (`v2`) untuk meminta dokumen JSON kebijakan menggunakan perintah berikut.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav menyimpan dokumen JSON kebijakan yang dikembalikan di `Document` bidang array. `PolicyVersion` Dalam dokumen kebijakan, Arnav mencari tindakan yang dalam namespace `ec2`. Jika tidak ada tindakan dari namespace lain yang tersisa dalam kebijakan, maka dia memisahkan kebijakan dari identitas yang terdampak (pengguna, grup, dan peran). Kemudian, dia menghapus kebijakan tersebut. Dalam hal ini, kebijakan tersebut mencakup layanan Amazon DynamoDB dan Amazon S3. Jadi Arnav menghapus EC2 tindakan Amazon dari dokumen dan menyimpan perubahannya. Kemudian, ia menggunakan perintah berikut untuk memperbarui kebijakan dengan menggunakan versi baru berkas tersebut dan menetapkan versi tersebut sebagai versi kebijakan default.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

`ExamplePolicy1`Kebijakan ini sekarang diperbarui untuk menghapus akses ke EC2 layanan Amazon yang tidak perlu.

Skenario IAM lainnya

Informasi tentang kapan IAM sumber daya (pengguna, grup, peran, atau kebijakan) terakhir kali mencoba mengakses layanan dapat membantu Anda ketika Anda menyelesaikan salah satu tugas berikut:

- Kebijakan – Menyunting kebijakan yang dikelola pelanggan atau kebijakan selaras yang sudah ada untuk menghapus izin
- Kebijakan – Mengonversi kebijakan selaras ke kebijakan terkelola dan kemudian menghapusnya
- Kebijakan – Menambahkan penolakan secara eksplisit ke kebijakan yang sudah ada
- Policies – Melepas kebijakan terkelola dari suatu identitas (pengguna, grup, atau peran)
- Entitas – Atur batas izin untuk mengendalikan perizinan maksimum yang dapat dimiliki oleh entitas (pengguna atau peran)
- Kelompok – Menghapus pengguna dari grup

Menggunakan informasi untuk memperbaiki izin unit organisasi.

Anda dapat menggunakan informasi yang diakses terakhir untuk memperbaiki izin unit organisasi (OU) di AWS Organizations.

Misalnya, John Stiles adalah seorang AWS Organizations administrator. Dia bertanggung jawab untuk memastikan bahwa orang-orang di perusahaan Akun AWS tidak memiliki izin berlebih. Sebagai bagian dari audit keamanan berkala, ia meninjau izin dari organisasinya. OU `Development`-nya berisikan akun yang sering digunakan untuk menguji layanan AWS baru. John memutuskan untuk secara berkala meninjau laporan bagi layanan yang belum diakses dalam lebih dari 180 hari. Kemudian, ia menghapus izin bagi anggota OU untuk mengakses layanan tersebut.

John masuk ke IAM konsol menggunakan kredensi akun manajemennya. Di IAM konsol, ia menemukan data Organizations untuk `Development` OU. Dia meninjau tabel laporan akses Layanan dan melihat dua AWS layanan yang belum diakses lebih dari periode yang diinginkannya selama 180 hari. Dia ingat menambahkan izin untuk tim pengembangan untuk mengakses Amazon Lex dan. AWS Database Migration Service John menghubungi tim pengembangan dan mengonfirmasi bahwa mereka tidak lagi memiliki kepentingan bisnis untuk menguji layanan ini.

John sekarang siap bertindak atas informasi yang terakhir diakses. Dia memilih Edit AWS Organizations dan diingatkan bahwa melekat pada beberapa entitas. SCP Dia memilih Lanjutkan. Pada tahun AWS Organizations, ia meninjau target untuk mempelajari entitas Organizations mana yang SCP dilampirkan. Semua entitas berada dalam OU `Development`.

John memutuskan untuk menolak akses ke Amazon Lex dan AWS Database Migration Service tindakan di `NewServiceTest`SCP. Tindakan ini menghapus akses yang tidak perlu ke layanan.

IAMtindakan terakhir diakses layanan informasi dan tindakan

Tabel berikut mencantumkan AWS layanan untuk IAMtindakan yang terakhir diakses informasi ditampilkan. Untuk daftar tindakan di setiap layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan di Referensi Otorisasi Layanan.

Layanan	Awalan layanan
AWS Identity and Access Management Penganalisis Akses	akses-analyzer
AWS Account Management	akun
AWS Certificate Manager	acm
Alur Kerja Terkelola Amazon untuk Apache Airflow	aliran udara
AWS Amplify	amplify
AWS Amplify Pembuat UI	amplifyuibuilder
Amazon AppIntegrations	integrasi aplikasi
AWS AppConfig	appconfig
Amazon AppFlow	Appflow
AWS Profiler Biaya Aplikasi	application-cost-profiler
Wawasan CloudWatch Aplikasi Amazon	wawasan aplikasi
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Layanan Terkelola Amazon untuk Prometheus	aps

Layanan	Awalan layanan
Amazon Athena	Athena
AWS Audit Manager	Pengelola Audit
AWS Auto Scaling	penskalaan otomatis
AWS Marketplace	aws-marketplace
AWS Backup	pencadangan
AWS Batch	batch
Amazon Braket	braket
AWS Budgets	anggaran
AWS Cloud9	awan9
AWS CloudFormation	pembentukan awan
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	kodeartefak
AWS CodeDeploy	penyebaran kode
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Peninjau Amazon	pengulas codeguru

Layanan	Awalan layanan
AWS CodePipeline	codepipeline
AWS CodeStar	bintang kode
AWS CodeStar Pemberitahuan	pemberitahuan bintang kode
Identitas Amazon Cognito	kognito-identitas
Kumpulan pengguna Amazon Cognito	cognito-idp
Sinkronisasi Amazon Cognito	sinkronis asi kognito
Amazon Comprehend Medical	memahami-medis
AWS Compute Optimizer	pengoptim al komputasi
AWS Config	config
Amazon Connect	hubungkan
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	pertukaran data
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
DevOpsGuru Amazon	devops-guru
AWS Direct Connect	DirectConnect
Amazon Data Lifecycle Manager	dlm

Layanan	Awalan layanan
AWS Database Migration Service	dms
Cluster Elastis Amazon DocumentDB	docdb-elastis
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Publik	ecr-publik
Layanan Kontainer Elastis Amazon	ecs
Layanan Amazon Elastic Kubernetes	eks
Amazon Elastic Inference	inferensi elastis
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	tangkai beanstalk
Sistem File Elastis Amazon	sistem file elastis
Elastic Load Balancing	elasticlo adbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR di EKS (EMRWadah)	kontainer emr
Amazon Tanpa EMR Server	emr-tanpa server
OpenSearch Layanan Amazon	es
Amazon EventBridge	peristiwa
Amazon CloudWatch Terbukti	ternyata

Layanan	Awalan layanan
Amazon FinSpace	ruang finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	detektor penipuan
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	gletser
Grafana yang Dikelola Amazon	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	stasiun tanah
Amazon GuardDuty	tugas jaga
AWS HealthLake	healthlake
Honeycode Amazon	kode madu
AWS Identity and Access Management	iam
AWS Toko Identitas	toko identitas
EC2Image Builder	imagebuilder
Amazon Inspector Klasik	inspektur
Amazon Inspector	inspektor2

Layanan	Awalan layanan
AWS IoT	iot
AWS IoT Analytics	iotanalitik
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	pembuat iottwinmaker
AWS IoT Wireless	iotwireless
Layanan Video Interaktif Amazon	ivs
Obrolan Layanan Video Interaktif Amazon	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalitik
AWS Key Management Service	km
AWS Lambda	lambda
Amazon Lex	lex
AWS License Manager Manajer Langganan Linux	license-manager-linux-subscriptions

Layanan	Awalan layanan
Amazon Lightsail	lightsail
CloudWatch Log Amazon	log
Amazon Lookout for Equipment	peralatan pencarian
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Blockchain yang Dikelola Amazon	terkelolablockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
AWS Rekomendasi Strategi Migrasi Hub	Migration hub-strategi
Amazon Pinpoint	penargetan seluler
Amazon MQ	mq
AWS Network Manager	manajer jaringan

Layanan	Awalan layanan
Studio Amazon Nimble	gesit
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	pos terdepan
AWS Organizations	organisasi
AWS Panorama	panorama
AWS Performance Insights	pi
EventBridgePipa Amazon	pipa
Amazon Polly	polly
Profil Pelanggan Amazon Connect	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Tempat Sampah Daur Ulang	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
Data Pergeseran Merah Amazon API	data pergesera n merah
AWS Migration Hub Refactor Spaces	ruang refaktor
Amazon Rekognition	rekognisi
AWS Resilience Hub	resiliencehub

Layanan	Awalan layanan
Penjelajah Sumber Daya AWS	sumber daya-penjelajah-2
AWS Resource Groups	kelompok sumber daya
AWS RoboMaker	pembuat robomak
AWS Identity and Access Management Peran Di Mana Saja	peranandi mana saja
Rute Amazon 53	route53
Amazon Route 53 Kontrol Pemulihan	rute53- recovery-control-config
Kesiapan Pemulihan Amazon Route 53	route53-p emulihan-kesiapan
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Layanan Penyimpanan Sederhana Amazon	s3
Amazon S3 di Outposts	pos terdepan s3
Kemampuan SageMaker geospasial Amazon	sagemaker-geospasial
Savings Plans	savingsplans
EventBridgeSkema Amazon	skema
Amazon SimpleDB	sdb
AWS Secrets Manager	manajer rahasia
AWS Security Hub	securityhub

Layanan	Awalan layanan
Danau Keamanan Amazon	Securitylake
AWS Serverless Application Repository	repo tanpa server
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Layanan Email Amazon Sederhana	ses
AWS Shield	melindungi
AWS Signer	penandatangan
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint SMS dan Layanan Suara	sms-suara
AWS Snowball	bola salju
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	insiden ssm
Manajer Sistem AWS untuk SAP	ssm-sap
AWS Step Functions	negara
AWS Security Token Service	sts
Layanan Alur Kerja Sederhana Amazon	swf
Amazon CloudWatch Synthetics	sintetis

Layanan	Awalan layanan
AWS Resource Groups Tagging API	tanda
Amazon Texttract	textract
Amazon Timestream	aliran waktu
AWS Pembangun Jaringan Telco	tnb
Amazon Transcribe	mentranskripsikan
AWS Transfer Family	transfer
Amazon Translate	menerjemahkan
ID Suara Amazon Connect	voiceid
VPCKisi Amazon	vpc-kisi
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Kebijaksanaan Amazon Connect	kebijaksanaan
Amazon WorkLink	tautan kerja
Amazon WorkSpaces	ruang kerja
AWS X-Ray	xray

Tindakan untuk tindakan informasi yang terakhir diakses

Tabel berikut mencantumkan tindakan untuk tindakan yang terakhir diakses informasi yang tersedia.

Awalan layanan	Tindakan
akses-analyzer	akses-analyzer: ApplyArchiveRule akses-analyzer: CancelPolicyGeneration

Awalan layanan	Tindakan
	akses-analyzer: CheckAccessNotGranted
	akses-analyzer: CheckNoNewAccess
	akses-analyzer: CheckNoPublicAccess
	akses-analyzer: CreateAccessPreview
	akses-analyzer: CreateAnalyzer
	akses-analyzer: CreateArchiveRule
	akses-analyzer: DeleteAnalyzer
	akses-analyzer: DeleteArchiveRule
	akses-analyzer: GenerateFindingRecommendation
	akses-analyzer: GetAccessPreview
	akses-analyzer: GetAnalyzedResource
	akses-analyzer: GetAnalyzer
	akses-analyzer: GetArchiveRule
	akses-analyzer: GetFinding
	akses-analyzer: GetFindingRecommendation
	akses-analyzer: GetGeneratedPolicy
	akses-analyzer: ListAccessPreviewFindings
	akses-analyzer: ListAccessPreviews
	akses-analyzer: ListAnalyzedResources
	akses-analyzer: ListAnalyzers
	akses-analyzer: ListArchiveRules

Awalan layanan	Tindakan
	akses-analyzer: ListFindings akses-analyzer: ListPolicyGenerations akses-analyzer: StartPolicyGeneration akses-analyzer: StartResourceScan akses-analyzer: UpdateArchiveRule akses-analyzer: UpdateFindings akses-analyzer: ValidatePolicy
akun	akun: AcceptPrimaryEmailUpdate akun: DeleteAlternateContact akun: DisableRegion akun: EnableRegion akun: GetAlternateContact akun: GetContactInformation akun: GetPrimaryEmail akun: GetRegionOptStatus akun: ListRegions akun: PutAlternateContact akun: PutContactInformation akun: StartPrimaryEmailUpdate

Awalan layanan	Tindakan
acm	ACM: DeleteCertificate ACM: DescribeCertificate ACM: ExportCertificate ACM: GetAccountConfiguration ACM: GetCertificate ACM: ImportCertificate ACM: ListCertificates ACM: PutAccountConfiguration ACM: RenewCertificate ACM: RequestCertificate ACM: ResendValidationEmail ACM: UpdateCertificateOptions
aliran udara	aliran udara: CreateCliToken aliran udara: CreateEnvironment aliran udara: CreateWebLoginToken aliran udara: DeleteEnvironment aliran udara: GetEnvironment aliran udara: ListEnvironments aliran udara: PublishMetrics aliran udara: UpdateEnvironment

Awalan layanan	Tindakan
amplify	memperkuat: CreateApp
	memperkuat: CreateBackendEnvironment
	memperkuat: CreateBranch
	memperkuat: CreateDeployment
	memperkuat: CreateDomainAssociation
	memperkuat: CreateWebHook
	memperkuat: DeleteApp
	memperkuat: DeleteBackendEnvironment
	memperkuat: DeleteBranch
	memperkuat: DeleteDomainAssociation
	memperkuat: DeleteJob
	memperkuat: DeleteWebHook
	memperkuat: GenerateAccessLogs
	memperkuat: GetApp
	memperkuat: GetArtifactUrl
	memperkuat: GetBackendEnvironment
	memperkuat: GetBranch
	memperkuat: GetDomainAssociation
	memperkuat: GetJob
	memperkuat: GetWebHook
	memperkuat: ListApps

Awalan layanan	Tindakan
	memperkuat: ListArtifacts
	memperkuat: ListBackendEnvironments
	memperkuat: ListBranches
	memperkuat: ListDomainAssociations
	memperkuat: ListJobs
	memperkuat: ListWebHooks
	memperkuat: StartDeployment
	memperkuat: StartJob
	memperkuat: StopJob
	memperkuat: UpdateApp
	memperkuat: UpdateBranch
	memperkuat: UpdateDomainAssociation
	memperkuat: UpdateWebHook

Awalan layanan	Tindakan
amplifyuibuilder	amplifyuibuilder: CreateComponent amplifyuibuilder: CreateForm amplifyuibuilder: CreateTheme amplifyuibuilder: DeleteComponent amplifyuibuilder: DeleteForm amplifyuibuilder: DeleteTheme amplifyuibuilder: ExportComponents amplifyuibuilder: ExportThemes amplifyuibuilder: GetCodegenJob amplifyuibuilder: ListCodegenJobs amplifyuibuilder: ListComponents amplifyuibuilder: ListForms amplifyuibuilder: ListThemes amplifyuibuilder: ResetMetadataFlag amplifyuibuilder: StartCodegenJob amplifyuibuilder: UpdateComponent amplifyuibuilder: UpdateForm amplifyuibuilder: UpdateTheme

Awalan layanan	Tindakan
integrasi aplikasi	integrasi aplikasi: CreateApplication
	integrasi aplikasi: CreateDataIntegration
	integrasi aplikasi: CreateDataIntegrationAssociation
	integrasi aplikasi: CreateEventIntegration
	integrasi aplikasi: DeleteApplication
	integrasi aplikasi: DeleteDataIntegration
	integrasi aplikasi: DeleteEventIntegration
	integrasi aplikasi: GetApplication
	integrasi aplikasi: GetDataIntegration
	integrasi aplikasi: GetEventIntegration
	integrasi aplikasi: ListApplicationAssociations
	integrasi aplikasi: ListApplications
	integrasi aplikasi: ListDataIntegrationAssociations
	integrasi aplikasi: ListDataIntegrations
	integrasi aplikasi: ListEventIntegrationAssociations
	integrasi aplikasi: ListEventIntegrations
	integrasi aplikasi: UpdateApplication
	integrasi aplikasi: UpdateDataIntegration
	integrasi aplikasi: UpdateDataIntegrationAssociation
	integrasi aplikasi: UpdateEventIntegration

Awalan layanan	Tindakan
appconfig	appconfig: CreateApplication appconfig: CreateConfigurationProfile appconfig: CreateDeploymentStrategy appconfig: CreateEnvironment appconfig: CreateExtension appconfig: CreateExtensionAssociation appconfig: CreateHostedConfigurationVersion appconfig: DeleteApplication appconfig: DeleteConfigurationProfile appconfig: DeleteDeploymentStrategy appconfig: DeleteEnvironment appconfig: DeleteExtension appconfig: DeleteExtensionAssociation appconfig: DeleteHostedConfigurationVersion appconfig: GetAccountSettings appconfig: GetApplication appconfig: GetConfiguration appconfig: GetConfigurationProfile appconfig: GetDeployment appconfig: GetDeploymentStrategy appconfig: GetEnvironment

Awalan layanan	Tindakan
	appconfig: GetExtension appconfig: GetExtensionAssociation appconfig: GetHostedConfigurationVersion appconfig: ListApplications appconfig: ListConfigurationProfiles appconfig: ListDeployments appconfig: ListDeploymentStrategies appconfig: ListEnvironments appconfig: ListExtensionAssociations appconfig: ListExtensions appconfig: ListHostedConfigurationVersions appconfig: StartDeployment appconfig: StopDeployment appconfig: UpdateAccountSettings appconfig: UpdateApplication appconfig: UpdateConfigurationProfile appconfig: UpdateDeploymentStrategy appconfig: UpdateEnvironment appconfig: UpdateExtension appconfig: UpdateExtensionAssociation appconfig: ValidateConfiguration

Awalan layanan	Tindakan
Appflow	Appflow: CancelFlowExecutions
	Appflow: CreateConnectorProfile
	Appflow: CreateFlow
	Appflow: DeleteConnectorProfile
	Appflow: DeleteFlow
	Appflow: DescribeConnector
	Appflow: DescribeConnectorEntity
	Appflow: DescribeConnectorProfiles
	Appflow: DescribeConnectors
	Appflow: DescribeFlow
	Appflow: DescribeFlowExecutionRecords
	Appflow: ListConnectorEntities
	Appflow: ListConnectors
	Appflow: ListFlows
	Appflow: RegisterConnector
	Appflow: ResetConnectorMetadataCache
	Appflow: StartFlow
	Appflow: StopFlow
	Appflow: UnRegisterConnector
	Appflow: UpdateConnectorProfile
	Appflow: UpdateConnectorRegistration

Awalan layanan	Tindakan
	Appflow: UpdateFlow
application-cost-profiler	application-cost-profiler:DeleteReportDefinition application-cost-profiler:GetReportDefinition application-cost-profiler:ImportApplicationUsage application-cost-profiler:ListReportDefinitions application-cost-profiler:PutReportDefinition application-cost-profiler:UpdateReportDefinition

Awalan layanan	Tindakan
wawasan aplikasi	wawasan aplikasi: AddWorkload
	wawasan aplikasi: CreateApplication
	wawasan aplikasi: CreateComponent
	wawasan aplikasi: CreateLogPattern
	wawasan aplikasi: DeleteApplication
	wawasan aplikasi: DeleteComponent
	wawasan aplikasi: DeleteLogPattern
	wawasan aplikasi: DescribeApplication
	wawasan aplikasi: DescribeComponent
	wawasan aplikasi: DescribeComponentConfiguration
	wawasan aplikasi: DescribeComponentConfigurationRecommendation
	wawasan aplikasi: DescribeLogPattern
	wawasan aplikasi: DescribeObservation
	wawasan aplikasi: DescribeProblem
	wawasan aplikasi: DescribeProblemObservations
	wawasan aplikasi: DescribeWorkload
	wawasan aplikasi: ListApplications
	wawasan aplikasi: ListComponents
	wawasan aplikasi: ListConfigurationHistory
	wawasan aplikasi: ListLogPatterns

Awalan layanan	Tindakan
	wawasan aplikasi: ListLogPatternSets
	wawasan aplikasi: ListProblems
	wawasan aplikasi: ListWorkloads
	wawasan aplikasi: RemoveWorkload
	wawasan aplikasi: UpdateApplication
	wawasan aplikasi: UpdateComponent
	wawasan aplikasi: UpdateComponentConfiguration
	wawasan aplikasi: UpdateLogPattern
	wawasan aplikasi: UpdateWorkload

Awalan layanan	Tindakan
appmesh	appmesh: CreateGatewayRoute appmesh: CreateMesh appmesh: CreateRoute appmesh: CreateVirtualGateway appmesh: CreateVirtualNode appmesh: CreateVirtualRouter appmesh: CreateVirtualService appmesh: DeleteGatewayRoute appmesh: DeleteMesh appmesh: DeleteRoute appmesh: DeleteVirtualGateway appmesh: DeleteVirtualNode appmesh: DeleteVirtualRouter appmesh: DeleteVirtualService appmesh: DescribeGatewayRoute appmesh: DescribeMesh appmesh: DescribeRoute appmesh: DescribeVirtualGateway appmesh: DescribeVirtualNode appmesh: DescribeVirtualRouter appmesh: DescribeVirtualService

Awalan layanan	Tindakan
	appmesh: ListGatewayRoutes appmesh: ListMeshes appmesh: ListRoutes appmesh: ListVirtualGateways appmesh: ListVirtualNodes appmesh: ListVirtualRouters appmesh: ListVirtualServices appmesh: StreamAggregatedResources appmesh: UpdateGatewayRoute appmesh: UpdateMesh appmesh: UpdateRoute appmesh: UpdateVirtualGateway appmesh: UpdateVirtualNode appmesh: UpdateVirtualRouter appmesh: UpdateVirtualService

Awalan layanan	Tindakan
appstream	appstream: AssociateAppBlockBuilderAppBlock
	appstream: AssociateApplicationFleet
	appstream: AssociateApplicationToEntitlement
	appstream: AssociateFleet
	appstream: BatchAssociateUserStack
	appstream: BatchDisassociateUserStack
	appstream: CopyImage
	appstream: CreateAppBlock
	appstream: CreateAppBlockBuilder
	appstream: CreateAppBlockBuilderStreaming URL
	appstream: CreateApplication
	appstream: CreateDirectoryConfig
	appstream: CreateEntitlement
	appstream: CreateFleet
	appstream: CreateImageBuilder
	appstream: CreateImageBuilderStreaming URL
	appstream: CreateStack
	appstream: CreateStreaming URL
	appstream: CreateThemeForStack
	appstream: CreateUpdatedImage
	appstream: CreateUsageReportSubscription

Awalan layanan	Tindakan
	appstream: CreateUser
	appstream: DeleteAppBlock
	appstream: DeleteAppBlockBuilder
	appstream: DeleteApplication
	appstream: DeleteDirectoryConfig
	appstream: DeleteEntitlement
	appstream: DeleteFleet
	appstream: DeleteImage
	appstream: DeleteImageBuilder
	appstream: DeleteImagePermissions
	appstream: DeleteStack
	appstream: DeleteThemeForStack
	appstream: DeleteUsageReportSubscription
	appstream: DeleteUser
	appstream: DescribeAppBlockBuilderAppBlockAssociations
	appstream: DescribeAppBlockBuilders
	appstream: DescribeAppBlocks
	appstream: DescribeApplicationFleetAssociations
	appstream: DescribeApplications
	appstream: DescribeDirectoryConfigs
	appstream: DescribeEntitlements

Awalan layanan	Tindakan
	appstream: DescribeFleets
	appstream: DescribeImageBuilders
	appstream: DescribeImagePermissions
	appstream: DescribeImages
	appstream: DescribeSessions
	appstream: DescribeStacks
	appstream: DescribeThemeForStack
	appstream: DescribeUsageReportSubscriptions
	appstream: DescribeUsers
	appstream: DescribeUserStackAssociations
	appstream: DisableUser
	appstream: DisassociateAppBlockBuilderAppBlock
	appstream: DisassociateApplicationFleet
	appstream: DisassociateApplicationFromEntitlement
	appstream: DisassociateFleet
	appstream: EnableUser
	appstream: ExpireSession
	appstream: ListAssociatedFleets
	appstream: ListAssociatedStacks
	appstream: ListEntitledApplications
	appstream: StartAppBlockBuilder

Awalan layanan	Tindakan
	appstream: StartFleet
	appstream: StartImageBuilder
	appstream: StopAppBlockBuilder
	appstream: StopFleet
	appstream: StopImageBuilder
	appstream: UpdateAppBlockBuilder
	appstream: UpdateApplication
	appstream: UpdateDirectoryConfig
	appstream: UpdateEntitlement
	appstream: UpdateFleet
	appstream: UpdateImagePermissions
	appstream: UpdateStack
	appstream: UpdateThemeForStack

Awalan layanan	Tindakan
appsync	appsync: AssociateApi
	appsync: AssociateMergedGraphqlApi
	appsync: AssociateSourceGraphqlApi
	appsync: CreateApiCache
	appsync: CreateApiKey
	appsync: CreateDataSource
	appsync: CreateDomainName
	appsync: CreateFunction
	appsync: CreateGraphqlApi
	appsync: CreateResolver
	appsync: CreateType
	appsync: DeleteApiCache
	appsync: DeleteApiKey
	appsync: DeleteDataSource
	appsync: DeleteDomainName
	appsync: DeleteFunction
	appsync: DeleteGraphqlApi
	appsync: DeleteResolver
	appsync: DeleteType
	appsync: DisassociateApi
	appsync: DisassociateMergedGraphqlApi

Awalan layanan	Tindakan
	appsync: DisassociateSourceGraphqlApi appsync: EvaluateCode appsync: EvaluateMappingTemplate appsync: FlushApiCache appsync: GetApiAssociation appsync: GetApiCache appsync: GetDataSource appsync: GetDataSourceIntrospection appsync: GetDomainName appsync: GetFunction appsync: GetGraphqlApi appsync: GetGraphqlApiEnvironmentVariables appsync: GetIntrospectionSchema appsync: GetResolver appsync: GetSchemaCreationStatus appsync: GetSourceApiAssociation appsync: GetType appsync: ListApiKeys appsync: ListDataSources appsync: ListDomainNames appsync: ListFunctions

Awalan layanan	Tindakan
	appsync: ListGraphqlApis appsync: ListResolvers appsync: ListResolversByFunction appsync: ListSourceApiAssociations appsync: ListTypes appsync: ListTypesByAssociation appsync: PutGraphqlApiEnvironmentVariables appsync: StartDataSourceIntrospection appsync: StartSchemaCreation appsync: StartSchemaMerge appsync: UpdateApiCache appsync: UpdateApiKey appsync: UpdateDataSource appsync: UpdateDomainName appsync: UpdateFunction appsync: UpdateGraphqlApi appsync: UpdateResolver appsync: UpdateSourceApiAssociation appsync: UpdateType

Awalan layanan	Tindakan
aps	aps: CreateAlertManagerDefinition
	aps: CreateLoggingConfiguration
	aps: CreateRuleGroupsNamespace
	aps: CreateScraper
	aps: CreateWorkspace
	aps: DeleteAlertManagerDefinition
	aps: DeleteLoggingConfiguration
	aps: DeleteRuleGroupsNamespace
	aps: DeleteScraper
	aps: DeleteWorkspace
	aps: DescribeAlertManagerDefinition
	aps: DescribeLoggingConfiguration
	aps: DescribeRuleGroupsNamespace
	aps: DescribeScraper
	aps: DescribeWorkspace
	aps: GetDefaultScraperConfiguration
	aps: ListRuleGroupsNamespaces
	aps: ListScrapers
	aps: ListWorkspaces
	aps: PutAlertManagerDefinition
	aps: PutRuleGroupsNamespace

Awalan layanan	Tindakan
	aps: UpdateLoggingConfiguration aps: UpdateWorkspaceAlias

Awalan layanan	Tindakan
Athena	Athena: BatchGetNamedQuery
	Athena: BatchGetPreparedStatement
	Athena: BatchGetQueryExecution
	Athena: CancelCapacityReservation
	Athena: CreateCapacityReservation
	Athena: CreateDataCatalog
	Athena: CreateNamedQuery
	Athena: CreateNotebook
	Athena: CreatePreparedStatement
	Athena: CreatePresignedNotebookUrl
	Athena: CreateWorkGroup
	Athena: DeleteCapacityReservation
	Athena: DeleteDataCatalog
	Athena: DeleteNamedQuery
	Athena: DeleteNotebook
	Athena: DeletePreparedStatement
	Athena: DeleteWorkGroup
	Athena: ExportNotebook
	Athena: GetCalculationExecution
	Athena: GetCalculationExecutionCode
	Athena: GetCalculationExecutionStatus

Awalan layanan	Tindakan
	Athena: GetCapacityAssignmentConfiguration
	Athena: GetCapacityReservation
	Athena: GetDatabase
	Athena: GetDataCatalog
	Athena: GetNamedQuery
	Athena: GetNotebookMetadata
	Athena: GetPreparedStatement
	Athena: GetQueryExecution
	Athena: GetQueryResults
	Athena: GetQueryResultsStream
	Athena: GetQueryRuntimeStatistics
	Athena: GetSession
	Athena: GetSessionStatus
	Athena: GetTableMetadata
	Athena: GetWorkGroup
	Athena: ImportNotebook
	Athena: ListApplication DPUSizes
	Athena: ListCalculationExecutions
	Athena: ListCapacityReservations
	Athena: ListDatabases
	Athena: ListDataCatalogs

Awalan layanan	Tindakan
	Athena: ListEngineVersions
	Athena: ListExecutors
	Athena: ListNamedQueries
	Athena: ListNotebookMetadata
	Athena: ListNotebookSessions
	Athena: ListPreparedStatements
	Athena: ListQueryExecutions
	Athena: ListSessions
	Athena: ListTableMetadata
	Athena: ListWorkGroups
	Athena: PutCapacityAssignmentConfiguration
	Athena: StartCalculationExecution
	Athena: StartQueryExecution
	Athena: StartSession
	Athena: StopCalculationExecution
	Athena: StopQueryExecution
	Athena: TerminateSession
	Athena: UpdateCapacityReservation
	Athena: UpdateDataCatalog
	Athena: UpdateNamedQuery
	Athena: UpdateNotebook

Awalan layanan	Tindakan
	Athena: UpdateNotebookMetadata Athena: UpdatePreparedStatement Athena: UpdateWorkGroup

Awalan layanan	Tindakan
Pengelola Audit	Manajer Audit: AssociateAssessmentReportEvidenceFolder
	Manajer Audit: BatchAssociateAssessmentReportEvidence
	Manajer Audit: BatchCreateDelegationByAssessment
	Manajer Audit: BatchDeleteDelegationByAssessment
	Manajer Audit: BatchDisassociateAssessmentReportEvidence
	Manajer Audit: BatchImportEvidenceToAssessmentControl
	Manajer Audit: CreateAssessment
	Manajer Audit: CreateAssessmentFramework
	Manajer Audit: CreateAssessmentReport
	Manajer Audit: CreateControl
	Manajer Audit: DeleteAssessment
	Manajer Audit: DeleteAssessmentFramework
	Manajer Audit: DeleteAssessmentFrameworkShare
	Manajer Audit: DeleteAssessmentReport
	Manajer Audit: DeleteControl
	Manajer Audit: DeregisterAccount
	Manajer Audit: DeregisterOrganizationAdminAccount
	Manajer Audit: DisassociateAssessmentReportEvidenceFolder
	Manajer Audit: GetAccountStatus
	Manajer Audit: GetAssessment
	Manajer Audit: GetAssessmentFramework

Awalan layanan	Tindakan
	Manajer Audit: GetAssessmentReportUrl
	Manajer Audit: GetChangeLogs
	Manajer Audit: GetControl
	Manajer Audit: GetDelegations
	Manajer Audit: GetEvidence
	Manajer Audit: GetEvidenceByEvidenceFolder
	Manajer Audit: GetEvidenceFileUploadUrl
	Manajer Audit: GetEvidenceFolder
	Manajer Audit: GetEvidenceFoldersByAssessment
	Manajer Audit: GetEvidenceFoldersByAssessmentControl
	Manajer Audit: GetInsights
	Manajer Audit: GetInsightsByAssessment
	Manajer Audit: GetOrganizationAdminAccount
	Manajer Audit: GetServicesInScope
	Manajer Audit: GetSettings
	Manajer Audit: ListAssessmentControlInsightsByControlDomain
	Manajer Audit: ListAssessmentFrameworks
	Manajer Audit: ListAssessmentFrameworkShareRequests
	Manajer Audit: ListAssessmentReports
	Manajer Audit: ListAssessments
	Manajer Audit: ListControlDomainInsights

Awalan layanan	Tindakan
	Manajer Audit: ListControlDomainInsightsByAssessment
	Manajer Audit: ListControlInsightsByControlDomain
	Manajer Audit: ListControls
	Manajer Audit: ListKeywordsForDataSource
	Manajer Audit: ListNotifications
	Manajer Audit: RegisterAccount
	Manajer Audit: RegisterOrganizationAdminAccount
	Manajer Audit: StartAssessmentFrameworkShare
	Manajer Audit: UpdateAssessment
	Manajer Audit: UpdateAssessmentControl
	Manajer Audit: UpdateAssessmentControlSetStatus
	Manajer Audit: UpdateAssessmentFramework
	Manajer Audit: UpdateAssessmentFrameworkShare
	Manajer Audit: UpdateAssessmentStatus
	Manajer Audit: UpdateControl
	Manajer Audit: UpdateSettings
	Manajer Audit: ValidateAssessmentReportIntegrity

Awalan layanan	Tindakan
penskalaan otomatis	penskalaan otomatis: AttachInstances penskalaan otomatis: AttachLoadBalancers penskalaan otomatis: AttachLoadBalancerTargetGroups penskalaan otomatis: AttachTrafficSources penskalaan otomatis: BatchDeleteScheduledAction penskalaan otomatis: BatchPutScheduledUpdateGroupAction penskalaan otomatis: CancelInstanceRefresh penskalaan otomatis: CompleteLifecycleAction penskalaan otomatis: CreateAutoScalingGroup penskalaan otomatis: CreateLaunchConfiguration penskalaan otomatis: DeleteAutoScalingGroup penskalaan otomatis: DeleteLaunchConfiguration penskalaan otomatis: DeleteLifecycleHook penskalaan otomatis: DeleteNotificationConfiguration penskalaan otomatis: DeletePolicy penskalaan otomatis: DeleteScheduledAction penskalaan otomatis: DeleteWarmPool penskalaan otomatis: DescribeAccountLimits penskalaan otomatis: DescribeAdjustmentTypes penskalaan otomatis: DescribeAutoScalingGroups penskalaan otomatis: DescribeAutoScalingInstances

Awalan layanan	Tindakan
	penskalaan otomatis: DescribeAutoScalingNotificationTypes
	penskalaan otomatis: DescribeInstanceRefreshes
	penskalaan otomatis: DescribeLaunchConfigurations
	penskalaan otomatis: DescribeLifecycleHooks
	penskalaan otomatis: DescribeLifecycleHookTypes
	penskalaan otomatis: DescribeLoadBalancers
	penskalaan otomatis: DescribeLoadBalancerTargetGroups
	penskalaan otomatis: DescribeMetricCollectionTypes
	penskalaan otomatis: DescribeNotificationConfigurations
	penskalaan otomatis: DescribePolicies
	penskalaan otomatis: DescribeScalingActivities
	penskalaan otomatis: DescribeScalingProcessTypes
	penskalaan otomatis: DescribeScheduledActions
	penskalaan otomatis: DescribeTerminationPolicyTypes
	penskalaan otomatis: DescribeTrafficSources
	penskalaan otomatis: DescribeWarmPool
	penskalaan otomatis: DetachInstances
	penskalaan otomatis: DetachLoadBalancers
	penskalaan otomatis: DetachLoadBalancerTargetGroups
	penskalaan otomatis: DetachTrafficSources
	penskalaan otomatis: DisableMetricsCollection

Awalan layanan	Tindakan
	penskalaan otomatis: EnableMetricsCollection penskalaan otomatis: EnterStandby penskalaan otomatis: ExecutePolicy penskalaan otomatis: ExitStandby penskalaan otomatis: GetPredictiveScalingForecast penskalaan otomatis: PutLifecycleHook penskalaan otomatis: PutNotificationConfiguration penskalaan otomatis: PutScalingPolicy penskalaan otomatis: PutScheduledUpdateGroupAction penskalaan otomatis: PutWarmPool penskalaan otomatis: RecordLifecycleActionHeartbeat penskalaan otomatis: ResumeProcesses penskalaan otomatis: RollbackInstanceRefresh penskalaan otomatis: SetDesiredCapacity penskalaan otomatis: SetInstanceHealth penskalaan otomatis: SetInstanceProtection penskalaan otomatis: StartInstanceRefresh penskalaan otomatis: SuspendProcesses penskalaan otomatis: TerminateInstanceInAutoScalingGroup penskalaan otomatis: UpdateAutoScalingGroup
aws-marketplace	aws-pasar: GetEntitlements

Awalan layanan	Tindakan
pencadangan	cadangan: CancelLegalHold cadangan: CreateBackupPlan cadangan: CreateBackupSelection cadangan: CreateBackupVault cadangan: CreateFramework cadangan: CreateLegalHold cadangan: CreateLogicallyAirGappedBackupVault cadangan: CreateReportPlan cadangan: CreateRestoreTestingPlan cadangan: CreateRestoreTestingSelection cadangan: DeleteBackupPlan cadangan: DeleteBackupSelection cadangan: DeleteBackupVault cadangan: DeleteBackupVaultAccessPolicy cadangan: DeleteBackupVaultLockConfiguration cadangan: DeleteBackupVaultNotifications cadangan: DeleteFramework cadangan: DeleteRecoveryPoint cadangan: DeleteReportPlan cadangan: DeleteRestoreTestingPlan cadangan: DeleteRestoreTestingSelection

Awalan layanan	Tindakan
	cadangan: DescribeBackupJob cadangan: DescribeBackupVault cadangan: DescribeCopyJob cadangan: DescribeFramework cadangan: DescribeGlobalSettings cadangan: DescribeProtectedResource cadangan: DescribeRecoveryPoint cadangan: DescribeRegionSettings cadangan: DescribeReportJob cadangan: DescribeReportPlan cadangan: DescribeRestoreJob cadangan: DisassociateRecoveryPoint cadangan: DisassociateRecoveryPointFromParent cadangan: ExportBackupPlanTemplate cadangan: GetBackupPlan cadangan: GetBackupPlanFrom JSON cadangan: GetBackupPlanFromTemplate cadangan: GetBackupSelection cadangan: GetBackupVaultAccessPolicy cadangan: GetBackupVaultNotifications cadangan: GetLegalHold

Awalan layanan	Tindakan
	cadangan: GetRecoveryPointRestoreMetadata
	cadangan: GetRestoreJobMetadata
	cadangan: GetRestoreTestingInferredMetadata
	cadangan: GetRestoreTestingPlan
	cadangan: GetRestoreTestingSelection
	cadangan: GetSupportedResourceTypes
	cadangan: ListBackupJobs
	cadangan: ListBackupJobSummaries
	cadangan: ListBackupPlans
	cadangan: ListBackupPlanTemplates
	cadangan: ListBackupPlanVersions
	cadangan: ListBackupSelections
	cadangan: ListBackupVaults
	cadangan: ListCopyJobs
	cadangan: ListCopyJobSummaries
	cadangan: ListFrameworks
	cadangan: ListLegalHolds
	cadangan: ListProtectedResources
	cadangan: ListRecoveryPointsByBackupVault
	cadangan: ListRecoveryPointsByLegalHold
	cadangan: ListRecoveryPointsByResource

Awalan layanan	Tindakan
	cadangan: ListReportJobs cadangan: ListReportPlans cadangan: ListRestoreJobs cadangan: ListRestoreJobsByProtectedResource cadangan: ListRestoreJobSummaries cadangan: ListRestoreTestingPlans cadangan: ListRestoreTestingSelections cadangan: PutBackupVaultAccessPolicy cadangan: PutBackupVaultLockConfiguration cadangan: PutBackupVaultNotifications cadangan: PutRestoreValidationResult cadangan: StartBackupJob cadangan: StartCopyJob cadangan: StartReportJob cadangan: StartRestoreJob cadangan: StopBackupJob cadangan: UpdateBackupPlan cadangan: UpdateFramework cadangan: UpdateGlobalSettings cadangan: UpdateRecoveryPointLifecycle cadangan: UpdateRegionSettings

Awalan layanan	Tindakan
	cadangan: UpdateReportPlan cadangan: UpdateRestoreTestingPlan cadangan: UpdateRestoreTestingSelection

Awalan layanan	Tindakan
batch	batch: CancelJob
	batch: CreateComputeEnvironment
	batch: CreateJobQueue
	batch: CreateSchedulingPolicy
	batch: DeleteComputeEnvironment
	batch: DeleteJobQueue
	batch: DeleteSchedulingPolicy
	batch: DeregisterJobDefinition
	batch: DescribeComputeEnvironments
	batch: DescribeJobDefinitions
	batch: DescribeJobQueues
	batch: DescribeJobs
	batch: DescribeSchedulingPolicies
	batch: GetJobQueueSnapshot
	batch: ListJobs
	batch: ListSchedulingPolicies
	batch: RegisterJobDefinition
	batch: SubmitJob
	batch: TerminateJob
	batch: UpdateComputeEnvironment
	batch: UpdateJobQueue

Awalan layanan	Tindakan
	batch: UpdateSchedulingPolicy
braket	Braket: CancelJob Braket: CancelQuantumTask Braket: CreateJob Braket: CreateQuantumTask Braket: GetDevice Braket: GetJob Braket: GetQuantumTask Braket: SearchDevices Braket: SearchJobs Braket: SearchQuantumTasks

Awalan layanan	Tindakan
anggaran	anggaran: ModifyBudget
	anggaran: CreateBudgetAction
	anggaran: ModifyBudget
	anggaran: ModifyBudget
	anggaran: ModifyBudget
	anggaran: DeleteBudgetAction
	anggaran: ModifyBudget
	anggaran: ModifyBudget
	anggaran: ViewBudget
	anggaran: DescribeBudgetAction
	anggaran: DescribeBudgetActionHistories
	anggaran: DescribeBudgetActionsForAccount
	anggaran: DescribeBudgetActionsForBudget
	anggaran: ViewBudget
	anggaran: ViewBudget
	anggaran: ViewBudget
	anggaran: ViewBudget
	anggaran: ViewBudget
	anggaran: ExecuteBudgetAction
	anggaran: ModifyBudget
	anggaran: UpdateBudgetAction

Awalan layanan	Tindakan
	anggaran: ModifyBudget anggaran: ModifyBudget
awan9	awan9: CreateEnvironment EC2 awan9: CreateEnvironmentMembership awan9: DeleteEnvironment awan9: DeleteEnvironmentMembership awan9: DescribeEnvironmentMemberships awan9: DescribeEnvironments awan9: DescribeEnvironmentStatus awan9: ListEnvironments awan9: UpdateEnvironment awan9: UpdateEnvironmentMembership

Awalan layanan	Tindakan
pembentukan awan	pembentukan awan: BatchDescribeTypeConfigurations
	pembentukan awan: CancelUpdateStack
	pembentukan awan: ContinueUpdateRollback
	pembentukan awan: CreateChangeSet
	pembentukan awan: CreateGeneratedTemplate
	pembentukan awan: CreateStack
	pembentukan awan: CreateStackInstances
	pembentukan awan: CreateStackSet
	pembentukan awan: DeactivateType
	pembentukan awan: DeleteChangeSet
	pembentukan awan: DeleteGeneratedTemplate
	pembentukan awan: DeleteStack
	pembentukan awan: DeleteStackInstances
	pembentukan awan: DeleteStackSet
	pembentukan awan: DeregisterType
	pembentukan awan: DescribeAccountLimits
	pembentukan awan: DescribeChangeSet
	pembentukan awan: DescribeChangeSetHooks
	pembentukan awan: DescribeGeneratedTemplate
	pembentukan awan: DescribeOrganizationsAccess
	pembentukan awan: DescribePublisher

Awalan layanan	Tindakan
	pembentukan awan: DescribeResourceScan
	pembentukan awan: DescribeStackDriftDetectionStatus
	pembentukan awan: DescribeStackEvents
	pembentukan awan: DescribeStackInstance
	pembentukan awan: DescribeStackResource
	pembentukan awan: DescribeStackResourceDrifts
	pembentukan awan: DescribeStackResources
	pembentukan awan: DescribeStacks
	pembentukan awan: DescribeStackSet
	pembentukan awan: DescribeStackSetOperation
	pembentukan awan: DescribeType
	pembentukan awan: DescribeTypeRegistration
	pembentukan awan: DetectStackDrift
	pembentukan awan: DetectStackResourceDrift
	pembentukan awan: DetectStackSetDrift
	pembentukan awan: EstimateTemplateCost
	pembentukan awan: ExecuteChangeSet
	pembentukan awan: GetGeneratedTemplate
	pembentukan awan: GetStackPolicy
	pembentukan awan: GetTemplate
	pembentukan awan: GetTemplateSummary

Awalan layanan	Tindakan
	pembentukan awan: ImportStacksToStackSet
	pembentukan awan: ListChangeSets
	pembentukan awan: ListExports
	pembentukan awan: ListGeneratedTemplates
	pembentukan awan: ListImports
	pembentukan awan: ListResourceScanRelatedResources
	pembentukan awan: ListResourceScanResources
	pembentukan awan: ListResourceScans
	pembentukan awan: ListStackInstanceResourceDrifts
	pembentukan awan: ListStackInstances
	pembentukan awan: ListStackResources
	pembentukan awan: ListStackSetAutoDeploymentTargets
	pembentukan awan: ListStackSetOperationResults
	pembentukan awan: ListStackSetOperations
	pembentukan awan: ListStackSets
	pembentukan awan: ListTypeRegistrations
	pembentukan awan: ListTypes
	pembentukan awan: ListTypeVersions
	pembentukan awan: PublishType
	pembentukan awan: RecordHandlerProgress
	pembentukan awan: RegisterPublisher

Awalan layanan	Tindakan
	pembentukan awan: RegisterType
	pembentukan awan: RollbackStack
	pembentukan awan: SetStackPolicy
	pembentukan awan: SetTypeConfiguration
	pembentukan awan: SetTypeDefaultVersion
	pembentukan awan: SignalResource
	pembentukan awan: StartResourceScan
	pembentukan awan: StopStackSetOperation
	pembentukan awan: TestType
	pembentukan awan: UpdateGeneratedTemplate
	pembentukan awan: UpdateStack
	pembentukan awan: UpdateStackInstances
	pembentukan awan: UpdateStackSet
	pembentukan awan: UpdateTerminationProtection
	pembentukan awan: ValidateTemplate

Awalan layanan	Tindakan
cloudfront	cloudfront: AssociateAlias cloudfront: CreateCachePolicy cloudfront: CreateCloudFrontOriginAccessIdentity cloudfront: CreateContinuousDeploymentPolicy cloudfront: CreateFieldLevelEncryptionConfig cloudfront: CreateFieldLevelEncryptionProfile cloudfront: CreateFunction cloudfront: CreateInvalidation cloudfront: CreateKeyGroup cloudfront: CreateKeyValueStore cloudfront: CreateMonitoringSubscription cloudfront: CreateOriginAccessControl cloudfront: CreateOriginRequestPolicy cloudfront: CreatePublicKey cloudfront: CreateRealtimeLogConfig cloudfront: CreateResponseHeadersPolicy cloudfront: DeleteCachePolicy cloudfront: DeleteCloudFrontOriginAccessIdentity cloudfront: DeleteContinuousDeploymentPolicy cloudfront: DeleteDistribution cloudfront: DeleteFieldLevelEncryptionConfig

Awalan layanan	Tindakan
	cloudfront: DeleteFieldLevelEncryptionProfile cloudfront: DeleteFunction cloudfront: DeleteKeyGroup cloudfront: DeleteKeyValueStore cloudfront: DeleteMonitoringSubscription cloudfront: DeleteOriginAccessControl cloudfront: DeleteOriginRequestPolicy cloudfront: DeletePublicKey cloudfront: DeleteRealtimeLogConfig cloudfront: DeleteResponseHeadersPolicy cloudfront: DeleteStreamingDistribution cloudfront: DescribeFunction cloudfront: DescribeKeyValueStore cloudfront: GetCachePolicy cloudfront: GetCachePolicyConfig cloudfront: GetCloudFrontOriginAccessIdentity cloudfront: GetCloudFrontOriginAccessIdentityConfig cloudfront: GetContinuousDeploymentPolicy cloudfront: GetContinuousDeploymentPolicyConfig cloudfront: GetDistributionConfig cloudfront: GetFieldLevelEncryption

Awalan layanan	Tindakan
	cloudfront: GetFieldLevelEncryptionConfig cloudfront: GetFieldLevelEncryptionProfile cloudfront: GetFieldLevelEncryptionProfileConfig cloudfront: GetFunction cloudfront: GetInvalidation cloudfront: GetKeyGroup cloudfront: GetKeyGroupConfig cloudfront: GetMonitoringSubscription cloudfront: GetOriginAccessControl cloudfront: GetOriginAccessControlConfig cloudfront: GetOriginRequestPolicy cloudfront: GetOriginRequestPolicyConfig cloudfront: GetPublicKey cloudfront: GetPublicKeyConfig cloudfront: GetRealtimeLogConfig cloudfront: GetResponseHeadersPolicy cloudfront: GetResponseHeadersPolicyConfig cloudfront: GetStreamingDistribution cloudfront: GetStreamingDistributionConfig cloudfront: ListCachePolicies cloudfront: ListCloudFrontOriginAccessIdentities

Awalan layanan	Tindakan
	cloudfront: ListConflictingAliases cloudfront: ListContinuousDeploymentPolicies cloudfront: ListDistributions cloudfront: ListDistributionsByCachePolicyId cloudfront: ListDistributionsByKeyGroup cloudfront: ListDistributionsByOriginRequestPolicyId cloudfront: ListDistributionsByRealtimeLogConfig cloudfront: ListDistributionsByResponseHeadersPolicyId cloudfront: ListDistributionsByWeb ACLId cloudfront: ListFieldLevelEncryptionConfigs cloudfront: ListFieldLevelEncryptionProfiles cloudfront: ListFunctions cloudfront: ListInvalidations cloudfront: ListKeyGroups cloudfront: ListKeyValueStores cloudfront: ListOriginAccessControls cloudfront: ListOriginRequestPolicies cloudfront: ListPublicKeys cloudfront: ListRealtimeLogConfigs cloudfront: ListResponseHeadersPolicies cloudfront: ListStreamingDistributions

Awalan layanan	Tindakan
	cloudfront: PublishFunction cloudfront: TestFunction cloudfront: UpdateCachePolicy cloudfront: UpdateCloudFrontOriginAccessIdentity cloudfront: UpdateContinuousDeploymentPolicy cloudfront: UpdateDistribution cloudfront: UpdateFieldLevelEncryptionConfig cloudfront: UpdateFieldLevelEncryptionProfile cloudfront: UpdateFunction cloudfront: UpdateKeyGroup cloudfront: UpdateKeyValueStore cloudfront: UpdateOriginAccessControl cloudfront: UpdateOriginRequestPolicy cloudfront: UpdatePublicKey cloudfront: UpdateRealtimeLogConfig cloudfront: UpdateResponseHeadersPolicy

Awalan layanan	Tindakan
cloudhsm	cloudhsm: CreateHsm
	cloudhsm: DeleteBackup
	cloudhsm: DeleteHsm
	cloudhsm: DeleteResourcePolicy
	cloudhsm: DescribeBackups
	cloudhsm: DescribeClusters
	cloudhsm: GetResourcePolicy
	cloudhsm: InitializeCluster
	cloudhsm: ModifyBackupAttributes
	cloudhsm: ModifyCluster
	cloudhsm: PutResourcePolicy
	cloudhsm: RestoreBackup

Awalan layanan	Tindakan
cloudsearch	cloudsearch: BuildSuggesters cloudsearch: CreateDomain cloudsearch: DefineAnalysisScheme cloudsearch: DefineExpression cloudsearch: DefineIndexField cloudsearch: DefineSuggester cloudsearch: DeleteAnalysisScheme cloudsearch: DeleteDomain cloudsearch: DeleteExpression cloudsearch: DeleteIndexField cloudsearch: DeleteSuggester cloudsearch: DescribeAnalysisSchemes cloudsearch: DescribeAvailabilityOptions cloudsearch: DescribeDomainEndpointOptions cloudsearch: DescribeDomains cloudsearch: DescribeExpressions cloudsearch: DescribeIndexFields cloudsearch: DescribeScalingParameters cloudsearch: DescribeServiceAccessPolicies cloudsearch: DescribeSuggesters cloudsearch: IndexDocuments

Awalan layanan	Tindakan
	cloudsearch: ListDomainNames cloudsearch: UpdateAvailabilityOptions cloudsearch: UpdateDomainEndpointOptions cloudsearch: UpdateScalingParameters cloudsearch: UpdateServiceAccessPolicies

Awalan layanan	Tindakan
cloudtrail	cloudtrail: CancelQuery
	cloudtrail: CreateChannel
	cloudtrail: CreateEventDataStore
	cloudtrail: CreateTrail
	cloudtrail: DeleteChannel
	cloudtrail: DeleteEventDataStore
	cloudtrail: DeleteResourcePolicy
	cloudtrail: DeleteTrail
	cloudtrail: DeregisterOrganizationDelegatedAdmin
	cloudtrail: DescribeQuery
	cloudtrail: DescribeTrails
	cloudtrail: DisableFederation
	cloudtrail: GenerateQuery
	cloudtrail: GetChannel
	cloudtrail: GetEventDataStore
	cloudtrail: GetEventDataStoreData
	cloudtrail: GetEventSelectors
	cloudtrail: GetImport
	cloudtrail: GetInsightSelectors
	cloudtrail: GetQueryResults
	cloudtrail: GetResourcePolicy

Awalan layanan	Tindakan
	cloudtrail: GetTrail
	cloudtrail: GetTrailStatus
	cloudtrail: ListChannels
	cloudtrail: ListEventDataStores
	cloudtrail: ListImportFailures
	cloudtrail: ListImports
	cloudtrail: ListPublicKeys
	cloudtrail: ListQueries
	cloudtrail: ListTrails
	cloudtrail: LookupEvents
	cloudtrail: PutEventSelectors
	cloudtrail: PutInsightSelectors
	cloudtrail: PutResourcePolicy
	cloudtrail: RegisterOrganizationDelegatedAdmin
	cloudtrail: RestoreEventDataStore
	cloudtrail: StartEventDataStoreIngestion
	cloudtrail: StartImport
	cloudtrail: StartLogging
	cloudtrail: StartQuery
	cloudtrail: StopEventDataStoreIngestion
	cloudtrail: StopImport

Awalan layanan	Tindakan
	cloudtrail: StopLogging cloudtrail: UpdateChannel cloudtrail: UpdateEventDataStore cloudtrail: UpdateTrail

Awalan layanan	Tindakan
cloudwatch	jam tangan awan: DeleteAlarms jam tangan awan: DeleteAnomalyDetector jam tangan awan: DeleteDashboards jam tangan awan: DeleteInsightRules jam tangan awan: DeleteMetricStream jam tangan awan: DescribeAlarmHistory jam tangan awan: DescribeAlarms jam tangan awan: DescribeAlarmsForMetric jam tangan awan: DescribeAnomalyDetectors jam tangan awan: DescribeInsightRules jam tangan awan: DisableAlarmActions jam tangan awan: DisableInsightRules jam tangan awan: EnableAlarmActions jam tangan awan: EnableInsightRules jam tangan awan: GetDashboard jam tangan awan: GetInsightRuleReport jam tangan awan: GetMetricStatistics jam tangan awan: GetMetricStream jam tangan awan: ListDashboards jam tangan awan: ListManagedInsightRules jam tangan awan: ListMetricStreams

Awalan layanan	Tindakan
	jam tangan awan: PutAnomalyDetector jam tangan awan: PutCompositeAlarm jam tangan awan: PutDashboard jam tangan awan: PutInsightRule jam tangan awan: PutManagedInsightRules jam tangan awan: PutMetricAlarm jam tangan awan: PutMetricStream jam tangan awan: SetAlarmState jam tangan awan: StartMetricStreams jam tangan awan: StopMetricStreams

Awalan layanan	Tindakan
kodeartefak	codeartefak: AssociateExternalConnection
	codeartefak: CopyPackageVersions
	codeartefak: CreateDomain
	codeartefak: CreateRepository
	codeartefak: DeleteDomain
	codeartefak: DeleteDomainPermissionsPolicy
	codeartefak: DeletePackage
	codeartefak: DeletePackageVersions
	codeartefak: DeleteRepository
	codeartefak: DeleteRepositoryPermissionsPolicy
	codeartefak: DescribeDomain
	codeartefak: DescribePackage
	codeartefak: DescribePackageVersion
	codeartefak: DescribeRepository
	codeartefak: DisassociateExternalConnection
	codeartefak: DisposePackageVersions
	codeartefak: GetAssociatedPackageGroup
	codeartefak: GetAuthorizationToken
	codeartefak: GetDomainPermissionsPolicy
	codeartefak: GetPackageVersionAsset
	codeartefak: GetPackageVersionReadme

Awalan layanan	Tindakan
	codeartefak: GetRepositoryEndpoint
	codeartefak: GetRepositoryPermissionsPolicy
	codeartefak: ListDomains
	codeartefak: ListPackageGroups
	codeartefak: ListPackages
	codeartefak: ListPackageVersionAssets
	codeartefak: ListPackageVersionDependencies
	codeartefak: ListPackageVersions
	codeartefak: ListRepositories
	codeartefak: ListRepositoriesInDomain
	codeartefak: PublishPackageVersion
	codeartefak: PutDomainPermissionsPolicy
	codeartefak: PutPackageMetadata
	codeartefak: PutPackageOriginConfiguration
	codeartefak: PutRepositoryPermissionsPolicy
	codeartefak: ReadFromRepository
	codeartefak: UpdatePackageVersionsStatus
	codeartefak: UpdateRepository

Awalan layanan	Tindakan
penyebaran kode	penyebaran kode: BatchGetApplicationRevisions penyebaran kode: BatchGetApplications penyebaran kode: BatchGetDeploymentGroups penyebaran kode: BatchGetDeploymentInstances penyebaran kode: BatchGetDeployments penyebaran kode: BatchGetDeploymentTargets penyebaran kode: BatchGetOnPremisesInstances penyebaran kode: ContinueDeployment penyebaran kode: CreateApplication penyebaran kode: CreateDeployment penyebaran kode: CreateDeploymentConfig penyebaran kode: CreateDeploymentGroup penyebaran kode: DeleteApplication penyebaran kode: DeleteDeploymentConfig penyebaran kode: DeleteDeploymentGroup penyebaran kode: DeleteGitHubAccountToken penyebaran kode: DeleteResourcesByExternalId penyebaran kode: DeregisterOnPremisesInstance penyebaran kode: GetApplication penyebaran kode: GetApplicationRevision penyebaran kode: GetDeployment

Awalan layanan	Tindakan
	penyebaran kode: GetDeploymentConfig
	penyebaran kode: GetDeploymentGroup
	penyebaran kode: GetDeploymentInstance
	penyebaran kode: GetDeploymentTarget
	penyebaran kode: GetOnPremisesInstance
	penyebaran kode: ListApplicationRevisions
	penyebaran kode: ListApplications
	penyebaran kode: ListDeploymentConfigs
	penyebaran kode: ListDeploymentGroups
	penyebaran kode: ListDeploymentInstances
	penyebaran kode: ListDeployments
	penyebaran kode: ListDeploymentTargets
	penyebaran kode: ListGitHubAccountTokenNames
	penyebaran kode: ListOnPremisesInstances
	penyebaran kode: PutLifecycleEventHookExecutionStatus
	penyebaran kode: RegisterApplicationRevision
	penyebaran kode: RegisterOnPremisesInstance
	penyebaran kode: SkipWaitTimeForInstanceTermination
	penyebaran kode: StopDeployment
	penyebaran kode: UpdateApplication
	penyebaran kode: UpdateDeploymentGroup

Awalan layanan	Tindakan
codeguru-profiler	codeguru-profiler: AddNotificationChannels
	codeguru-profiler: BatchGetFrameMetricData
	codeguru-profiler: ConfigureAgent
	codeguru-profiler: CreateProfilingGroup
	codeguru-profiler: DeleteProfilingGroup
	codeguru-profiler: DescribeProfilingGroup
	codeguru-profiler: GetFindingsReportAccountSummary
	codeguru-profiler: GetNotificationConfiguration
	codeguru-profiler: GetPolicy
	codeguru-profiler: GetProfile
	codeguru-profiler: GetRecommendations
	codeguru-profiler: ListFindingsReports
	codeguru-profiler: ListProfileTimes
	codeguru-profiler: ListProfilingGroups
	codeguru-profiler: PutPermission
	codeguru-profiler: RemoveNotificationChannel
	codeguru-profiler: RemovePermission
	codeguru-profiler: SubmitFeedback
	codeguru-profiler: UpdateProfilingGroup

Awalan layanan	Tindakan
pengulas codeguru	pengulas codeguru: AssociateRepository pengulas codeguru: CreateCodeReview pengulas codeguru: DescribeCodeReview pengulas codeguru: DescribeRecommendationFeedback pengulas codeguru: DescribeRepositoryAssociation pengulas codeguru: DisassociateRepository pengulas codeguru: ListCodeReviews pengulas codeguru: ListRecommendationFeedback pengulas codeguru: ListRecommendations pengulas codeguru: ListRepositoryAssociations pengulas codeguru: PutRecommendationFeedback

Awalan layanan	Tindakan
codepipeline	codepipeline: AcknowledgeJob
	codepipeline: AcknowledgeThirdPartyJob
	codepipeline: CreateCustomActionType
	codepipeline: CreatePipeline
	codepipeline: DeleteCustomActionType
	codepipeline: DeletePipeline
	codepipeline: DeleteWebhook
	codepipeline: DeregisterWebhookWithThirdParty
	codepipeline: GetActionType
	codepipeline: GetJobDetails
	codepipeline: GetPipeline
	codepipeline: GetPipelineExecution
	codepipeline: GetPipelineState
	codepipeline: GetThirdPartyJobDetails
	codepipeline: ListActionExecutions
	codepipeline: ListActionTypes
	codepipeline: ListPipelineExecutions
	codepipeline: ListPipelines
	codepipeline: ListRuleExecutions
	codepipeline: ListRuleTypes
	codepipeline: ListWebhooks

Awalan layanan	Tindakan
	codepipeline: OverrideStageCondition
	codepipeline: PollForJobs
	codepipeline: PollForThirdPartyJobs
	codepipeline: PutActionRevision
	codepipeline: PutApprovalResult
	codepipeline: PutJobFailureResult
	codepipeline: PutJobSuccessResult
	codepipeline: PutThirdPartyJobFailureResult
	codepipeline: PutThirdPartyJobSuccessResult
	codepipeline: PutWebhook
	codepipeline: RegisterWebhookWithThirdParty
	codepipeline: RollbackStage
	codepipeline: StartPipelineExecution
	codepipeline: StopPipelineExecution
	codepipeline: UpdateActionType
	codepipeline: UpdatePipeline

Awalan layanan	Tindakan
bintang kode	bintang kode: AssociateTeamMember bintang kode: CreateProject bintang kode: CreateUserProfile bintang kode: DeleteProject bintang kode: DeleteUserProfile bintang kode: DescribeProject bintang kode: DescribeUserProfile bintang kode: DisassociateTeamMember bintang kode: ListProjects bintang kode: ListResources bintang kode: ListTeamMembers bintang kode: ListUserProfiles bintang kode: UpdateProject bintang kode: UpdateTeamMember bintang kode: UpdateUserProfile

Awalan layanan	Tindakan
pemberitahuan bintang kode	pemberitahuan bintang kode: CreateNotificationRule
	pemberitahuan bintang kode: DeleteNotificationRule
	pemberitahuan bintang kode: DeleteTarget
	pemberitahuan bintang kode: DescribeNotificationRule
	pemberitahuan bintang kode: ListEventTypes
	pemberitahuan bintang kode: ListNotificationRules
	pemberitahuan bintang kode: ListTargets
	Codestar-notifikasi:Berlangganan
	Codestar-notifications:berhenti berlangganan
	pemberitahuan bintang kode: UpdateNotificationRule

Awalan layanan	Tindakan
kognito-identitas	kognito-identitas: CreateIdentityPool kognito-identitas: DeleteIdentities kognito-identitas: DeleteIdentityPool kognito-identitas: DescribeIdentity kognito-identitas: DescribeIdentityPool kognito-identitas: GetIdentityPoolRoles kognito-identitas: ListIdentities kognito-identitas: ListIdentityPools kognito-identitas: LookupDeveloperIdentity kognito-identitas: MergeDeveloperIdentities kognito-identitas: SetIdentityPoolRoles kognito-identitas: UnlinkDeveloperIdentity kognito-identitas: UpdateIdentityPool

Awalan layanan	Tindakan
cognito-idp	cognito-idp: AddCustomAttributes
	cognito-idp: AdminAddUserToGroup
	cognito-idp: AdminConfirmSignUp
	cognito-idp: AdminCreateUser
	cognito-idp: AdminDeleteUser
	cognito-idp: AdminDeleteUserAttributes
	cognito-idp: AdminDisableProviderForUser
	cognito-idp: AdminDisableUser
	cognito-idp: AdminEnableUser
	cognito-idp: AdminForgetDevice
	cognito-idp: AdminGetDevice
	cognito-idp: AdminGetUser
	cognito-idp: AdminInitiateAuth
	cognito-idp: AdminLinkProviderForUser
	cognito-idp: AdminListDevices
	cognito-idp: AdminListGroupsForUser
	cognito-idp: AdminListUserAuthEvents
	cognito-idp: AdminRemoveUserFromGroup
	cognito-idp: AdminResetUserPassword
	cognito-idp: AdminRespondToAuthChallenge
	cognito-idp: AdminSetUser MFAPreference

Awalan layanan	Tindakan
	cognito-idp: AdminSetUserPassword
	cognito-idp: AdminSetUserSettings
	cognito-idp: AdminUpdateAuthEventFeedback
	cognito-idp: AdminUpdateDeviceStatus
	cognito-idp: AdminUpdateUserAttributes
	cognito-idp: AdminUserGlobalSignOut
	cognito-idp: AssociateSoftwareToken
	cognito-idp: ChangePassword
	cognito-idp: ConfirmDevice
	cognito-idp: ConfirmForgotPassword
	cognito-idp: ConfirmSignUp
	cognito-idp: CreateGroup
	cognito-idp: CreateIdentityProvider
	cognito-idp: CreateResourceServer
	cognito-idp: CreateUserImportJob
	cognito-idp: CreateUserPool
	cognito-idp: CreateUserPoolClient
	cognito-idp: CreateUserPoolDomain
	cognito-idp: DeleteGroup
	cognito-idp: DeleteIdentityProvider
	cognito-idp: DeleteResourceServer

Awalan layanan	Tindakan
	cognito-idp: DeleteUser
	cognito-idp: DeleteUserAttributes
	cognito-idp: DeleteUserPool
	cognito-idp: DeleteUserPoolClient
	cognito-idp: DeleteUserPoolDomain
	cognito-idp: DescribeIdentityProvider
	cognito-idp: DescribeResourceServer
	cognito-idp: DescribeRiskConfiguration
	cognito-idp: DescribeUserImportJob
	cognito-idp: DescribeUserPool
	cognito-idp: DescribeUserPoolClient
	cognito-idp: DescribeUserPoolDomain
	cognito-idp: ForgetDevice
	cognito-idp: ForgotPassword
	Cognito-IDP:G etCSVHeader
	cognito-idp: GetDevice
	cognito-idp: GetGroup
	cognito-idp: GetIdentityProviderByIdentifier
	cognito-idp: GetLogDeliveryConfiguration
	cognito-idp: GetSigningCertificate
	Cognito-IDP:G etUICustomization

Awalan layanan	Tindakan
	cognito-idp: GetUser
	cognito-idp: GetUserAttributeVerificationCode
	cognito-idp: GetUserPoolMfaConfig
	cognito-idp: GlobalSignOut
	cognito-idp: InitiateAuth
	cognito-idp: ListDevices
	cognito-idp: ListGroups
	cognito-idp: ListIdentityProviders
	cognito-idp: ListResourceServers
	cognito-idp: ListUserImportJobs
	cognito-idp: ListUserPoolClients
	cognito-idp: ListUserPools
	cognito-idp: ListUsers
	cognito-idp: ListUsersInGroup
	cognito-idp: ResendConfirmationCode
	cognito-idp: RespondToAuthChallenge
	cognito-idp: RevokeToken
	cognito-idp: SetLogDeliveryConfiguration
	cognito-idp: SetRiskConfiguration
	Cognito-IDP:S etUICustomization
	cognito-idp: SetUser MFAPreference

Awalan layanan	Tindakan
	cognito-idp: SetUserPoolMfaConfig
	cognito-idp: SetUserSettings
	cognito-idp: SignUp
	cognito-idp: StartUserImportJob
	cognito-idp: StopUserImportJob
	cognito-idp: UpdateAuthEventFeedback
	cognito-idp: UpdateDeviceStatus
	cognito-idp: UpdateGroup
	cognito-idp: UpdateIdentityProvider
	cognito-idp: UpdateResourceServer
	cognito-idp: UpdateUserAttributes
	cognito-idp: UpdateUserPool
	cognito-idp: UpdateUserPoolClient
	cognito-idp: UpdateUserPoolDomain
	cognito-idp: VerifySoftwareToken
	cognito-idp: VerifyUserAttribute

Awalan layanan	Tindakan
sinkronisasi kognito	sinkronisasi kognito: BulkPublish sinkronisasi kognito: DeleteDataset sinkronisasi kognito: DescribeDataset sinkronisasi kognito: DescribeIdentityPoolUsage sinkronisasi kognito: DescribeIdentityUsage sinkronisasi kognito: GetBulkPublishDetails sinkronisasi kognito: GetCognitoEvents sinkronisasi kognito: GetIdentityPoolConfiguration sinkronisasi kognito: ListDatasets sinkronisasi kognito: ListIdentityPoolUsage sinkronisasi kognito: ListRecords sinkronisasi kognito: RegisterDevice sinkronisasi kognito: SetCognitoEvents sinkronisasi kognito: SetIdentityPoolConfiguration sinkronisasi kognito: SubscribeToDataset sinkronisasi kognito: UnsubscribeFromDataset sinkronisasi kognito: UpdateRecords

Awalan layanan	Tindakan
memahami-medis	memahami medis: V2Job DescribeEntitiesDetection ICD1ComprehendMedical:Deskripsikan 0 CMInferenceJob PengetahuanMedis: D escribePHIDetection Job memahami medis: DescribeRxNormInferenceJob PengetahuanMedis: D escribeSNOMEDCTInference Job memahami medis: V2 DetectEntities ComprehendMedical: Deteksi PHI ICD1ComprehendMedical: Menyimpulkan 0CM memahami medis: InferRxNorm MemahamiHendMedical: Menyimpulkan SNOMEDCT memahami medis: V2Jobs ListEntitiesDetection ICD1ComprehendMedical:List 0 CMInferenceJobs istPHIDetectionComprehendMedical:L Jobs memahami medis: ListRxNormInferenceJobs istSNOMEDCTInferenceComprehendMedical:L Jobs memahami medis: V2Job StartEntitiesDetection MemahamiMedisICD1: Mulai 0 CMInferenceJob tartPHIDetectionComprehendMedical: s Job memahami medis: StartRxNormInferenceJob tartSNOMEDCTInferenceComprehendMedical: s Job memahami medis: V2Job StopEntitiesDetection

Awalan layanan	Tindakan
	MemahamiMedisICD1: Stop 0 CMInferenceJob topPHIDetectionComprehendMedical: s Job memahami medis: StopRxNormInferenceJob topSNOMEDCTInferenceComprehendMedical: s Job

Awalan layanan	Tindakan
pengoptimal komputasi	pengoptimal komputasi: DeleteRecommendationPreferences pengoptimal komputasi: DescribeRecommendationExportJobs pengoptimal komputasi: ExportAutoScalingGroupRecommendations Pengoptimal komputasixportEBSVolume: Rekomendasi E Pengoptimal komputasi: Ekspor EC2InstanceRecommendations Pengoptimal komputasixportECSService: Rekomendasi E pengoptimal komputasi: ExportLambdaFunctionRecommendations pengoptimal komputasi: ExportLicenseRecommendations Pengoptimal komputasixportRDSDatabase: Rekomendasi E Pengoptimal komputasi: Dapatkan EC2RecommendationP rojectedMetrics Pengoptimal komputasi: G etECSService RecommendationProj ectedMetrics pengoptimal komputasi: GetEffectiveRecommendationPreferences pengoptimal komputasi: GetEnrollmentStatus pengoptimal komputasi: GetEnrollmentStatusesForOrganization Pengoptimal komputasi: G etRDSDatabase RecommendationProj ectedMetrics pengoptimal komputasi: GetRecommendationPreferences pengoptimal komputasi: GetRecommendationSummaries pengoptimal komputasi: PutRecommendationPreferences pengoptimal komputasi: UpdateEnrollmentStatus

Awalan layanan	Tindakan
config	konfigurasi: BatchGetResourceConfig
	konfigurasi: DeleteAggregationAuthorization
	konfigurasi: DeleteConfigRule
	konfigurasi: DeleteConfigurationAggregator
	konfigurasi: DeleteConfigurationRecorder
	konfigurasi: DeleteConformancePack
	konfigurasi: DeleteDeliveryChannel
	konfigurasi: DeleteEvaluationResults
	konfigurasi: DeleteOrganizationConfigRule
	konfigurasi: DeleteOrganizationConformancePack
	konfigurasi: DeletePendingAggregationRequest
	konfigurasi: DeleteRemediationConfiguration
	konfigurasi: DeleteRemediationExceptions
	konfigurasi: DeleteResourceConfig
	konfigurasi: DeleteRetentionConfiguration
	konfigurasi: DeleteStoredQuery
	konfigurasi: DeliverConfigSnapshot
	konfigurasi: DescribeAggregateComplianceByConfigRules
	konfigurasi: DescribeAggregateComplianceByConformancePacks
	konfigurasi: DescribeAggregationAuthorizations
	konfigurasi: DescribeComplianceByConfigRule

Awalan layanan	Tindakan
	konfigurasi: DescribeComplianceByResource
	konfigurasi: DescribeConfigRuleEvaluationStatus
	konfigurasi: DescribeConfigRules
	konfigurasi: DescribeConfigurationAggregators
	konfigurasi: DescribeConfigurationAggregatorSourcesStatus
	konfigurasi: DescribeConfigurationRecorders
	konfigurasi: DescribeConfigurationRecorderStatus
	konfigurasi: DescribeConformancePackCompliance
	konfigurasi: DescribeConformancePacks
	konfigurasi: DescribeConformancePackStatus
	konfigurasi: DescribeDeliveryChannels
	konfigurasi: DescribeDeliveryChannelStatus
	konfigurasi: DescribeOrganizationConfigRules
	konfigurasi: DescribeOrganizationConfigRuleStatuses
	konfigurasi: DescribeOrganizationConformancePacks
	konfigurasi: DescribeOrganizationConformancePackStatuses
	konfigurasi: DescribePendingAggregationRequests
	konfigurasi: DescribeRemediationConfigurations
	konfigurasi: DescribeRemediationExceptions
	konfigurasi: DescribeRemediationExecutionStatus
	konfigurasi: DescribeRetentionConfigurations

Awalan layanan	Tindakan
	konfigurasi: GetComplianceDetailsByConfigRule
	konfigurasi: GetComplianceDetailsByResource
	konfigurasi: GetComplianceSummaryByConfigRule
	konfigurasi: GetComplianceSummaryByResourceType
	konfigurasi: GetConformancePackComplianceDetails
	konfigurasi: GetConformancePackComplianceSummary
	konfigurasi: GetCustomRulePolicy
	konfigurasi: GetDiscoveredResourceCounts
	konfigurasi: GetOrganizationConfigRuleDetailedStatus
	konfigurasi: GetOrganizationConformancePackDetailedStatus
	konfigurasi: GetOrganizationCustomRulePolicy
	konfigurasi: GetResourceConfigHistory
	konfigurasi: GetResourceEvaluationSummary
	konfigurasi: GetStoredQuery
	konfigurasi: ListConformancePackComplianceScores
	konfigurasi: ListDiscoveredResources
	konfigurasi: ListResourceEvaluations
	konfigurasi: ListStoredQueries
	konfigurasi: PutConfigRule
	konfigurasi: PutConfigurationAggregator
	konfigurasi: PutConfigurationRecorder

Awalan layanan	Tindakan
	konfigurasi: PutConformancePack
	konfigurasi: PutDeliveryChannel
	konfigurasi: PutEvaluations
	konfigurasi: PutExternalEvaluation
	konfigurasi: PutOrganizationConfigRule
	konfigurasi: PutOrganizationConformancePack
	konfigurasi: PutRemediationConfigurations
	konfigurasi: PutRemediationExceptions
	konfigurasi: PutResourceConfig
	konfigurasi: PutRetentionConfiguration
	konfigurasi: PutStoredQuery
	konfigurasi: SelectResourceConfig
	konfigurasi: StartConfigRulesEvaluation
	konfigurasi: StartConfigurationRecorder
	konfigurasi: StartRemediationExecution
	konfigurasi: StartResourceEvaluation
	konfigurasi: StopConfigurationRecorder

Awalan layanan	Tindakan
hubungkan	menghubungkan: ActivateEvaluationForm menghubungkan: AssociateApprovedOrigin menghubungkan: AssociateBot menghubungkan: AssociateDefaultVocabulary menghubungkan: AssociateFlow menghubungkan: AssociateInstanceStorageConfig menghubungkan: AssociateLambdaFunction menghubungkan: AssociateLexBot menghubungkan: AssociatePhoneNumberContactFlow menghubungkan: AssociateQueueQuickConnects menghubungkan: AssociateRoutingProfileQueues menghubungkan: AssociateSecurityKey menghubungkan: AssociateUserProficiencies menghubungkan: BatchGetFlowAssociation menghubungkan: BatchPutContact menghubungkan: ClaimPhoneNumber menghubungkan: CreateAgentStatus menghubungkan: CreateContactFlow menghubungkan: CreateContactFlowModule menghubungkan: CreateEvaluationForm menghubungkan: CreateHoursOfOperation

Awalan layanan	Tindakan
	menghubungkan: CreateInstance menghubungkan: CreateIntegrationAssociation menghubungkan: CreateParticipant menghubungkan: CreatePersistentContactAssociation menghubungkan: CreatePredefinedAttribute menghubungkan: CreatePrompt menghubungkan: CreateQueue menghubungkan: CreateQuickConnect menghubungkan: CreateRoutingProfile menghubungkan: CreateRule menghubungkan: CreateSecurityProfile menghubungkan: CreateTaskTemplate menghubungkan: CreateTrafficDistributionGroup menghubungkan: CreateUseCase menghubungkan: CreateUser menghubungkan: CreateUserHierarchyGroup menghubungkan: CreateView menghubungkan: CreateViewVersion menghubungkan: CreateVocabulary menghubungkan: DeactivateEvaluationForm menghubungkan: DeleteContactEvaluation

Awalan layanan	Tindakan
	menghubungkan: DeleteContactFlow menghubungkan: DeleteContactFlowModule menghubungkan: DeleteEvaluationForm menghubungkan: DeleteHoursOfOperation menghubungkan: DeleteInstance menghubungkan: DeleteIntegrationAssociation menghubungkan: DeletePredefinedAttribute menghubungkan: DeletePrompt menghubungkan: DeleteQueue menghubungkan: DeleteQuickConnect menghubungkan: DeleteRoutingProfile menghubungkan: DeleteRule menghubungkan: DeleteSecurityProfile menghubungkan: DeleteTaskTemplate menghubungkan: DeleteTrafficDistributionGroup menghubungkan: DeleteUseCase menghubungkan: DeleteUser menghubungkan: DeleteUserHierarchyGroup menghubungkan: DeleteView menghubungkan: DeleteVocabulary menghubungkan: DescribeAuthenticationProfile

Awalan layanan	Tindakan
	menghubungkan: DescribeContactEvaluation menghubungkan: DescribeEvaluationForm menghubungkan: DescribeInstanceAttribute menghubungkan: DescribeInstanceStorageConfig menghubungkan: DescribePhoneNumber menghubungkan: DescribeRule menghubungkan: DescribeTrafficDistributionGroup menghubungkan: DescribeUserHierarchyStructure menghubungkan: DescribeView menghubungkan: DescribeVocabulary menghubungkan: DisassociateApprovedOrigin menghubungkan: DisassociateBot menghubungkan: DisassociateFlow menghubungkan: DisassociateInstanceStorageConfig menghubungkan: DisassociateLambdaFunction menghubungkan: DisassociateLexBot menghubungkan: DisassociatePhoneNumberContactFlow menghubungkan: DisassociateQueueQuickConnects menghubungkan: DisassociateRoutingProfileQueues menghubungkan: DisassociateSecurityKey menghubungkan: DisassociateUserProficiencies

Awalan layanan	Tindakan
	menghubungkan: DismissUserContact menghubungkan: GetContactAttributes menghubungkan: GetCurrentMetricData menghubungkan: GetCurrentUserData menghubungkan: GetFederationToken menghubungkan: GetFlowAssociation menghubungkan: GetMetricData menghubungkan: GetMetricData V2 menghubungkan: GetPromptFile menghubungkan: GetTaskTemplate menghubungkan: GetTrafficDistribution menghubungkan: ImportPhoneNumber menghubungkan: ListApprovedOrigins menghubungkan: ListAuthenticationProfiles menghubungkan: ListBots menghubungkan: ListContactEvaluations menghubungkan: ListContactFlowModules menghubungkan: ListContactFlows menghubungkan: ListContactReferences menghubungkan: ListDefaultVocabularies menghubungkan: ListEvaluationForms

Awalan layanan	Tindakan
	menghubungkan: ListEvaluationFormVersions
	menghubungkan: ListFlowAssociations
	menghubungkan: ListHoursOfOperations
	menghubungkan: ListInstanceAttributes
	menghubungkan: ListInstanceStorageConfigs
	menghubungkan: ListIntegrationAssociations
	menghubungkan: ListLambdaFunctions
	menghubungkan: ListLexBots
	menghubungkan: ListPhoneNumbers
	menghubungkan: ListPhoneNumbers V2
	menghubungkan: ListPredefinedAttributes
	menghubungkan: ListPrompts
	menghubungkan: ListQueueQuickConnects
	menghubungkan: ListQueues
	menghubungkan: ListQuickConnects
	menghubungkan: ListRealtimeContactAnalysisSegments V2
	menghubungkan: ListRoutingProfileQueues
	menghubungkan: ListRoutingProfiles
	menghubungkan: ListRules
	menghubungkan: ListSecurityKeys
	menghubungkan: ListSecurityProfileApplications

Awalan layanan	Tindakan
	menghubungkan: ListSecurityProfilePermissions menghubungkan: ListSecurityProfiles menghubungkan: ListTaskTemplates menghubungkan: ListTrafficDistributionGroups menghubungkan: ListUseCases menghubungkan: ListUserHierarchyGroups menghubungkan: ListUsers menghubungkan: ListViews menghubungkan: ListViewVersions menghubungkan: MonitorContact menghubungkan: PauseContact menghubungkan: PutUserStatus menghubungkan: ReleasePhoneNumber menghubungkan: ReplicateInstance menghubungkan: ResumeContact menghubungkan: ResumeContactRecording menghubungkan: SearchAgentStatuses menghubungkan: SearchAvailablePhoneNumbers menghubungkan: SearchContactFlowModules menghubungkan: SearchContactFlows menghubungkan: SearchContacts

Awalan layanan	Tindakan
	menghubungkan: SearchHoursOfOperations menghubungkan: SearchPredefinedAttributes menghubungkan: SearchPrompts menghubungkan: SearchQueues menghubungkan: SearchQuickConnects menghubungkan: SearchRoutingProfiles menghubungkan: SearchSecurityProfiles menghubungkan: SearchUserHierarchyGroups menghubungkan: SearchVocabularies menghubungkan: SendChatIntegrationEvent menghubungkan: StartChatContact menghubungkan: StartContactEvaluation menghubungkan: StartContactRecording menghubungkan: StartContactStreaming menghubungkan: StartOutboundVoiceContact menghubungkan: StartTaskContact menghubungkan: StartWeb RTCContact menghubungkan: StopContact menghubungkan: StopContactRecording menghubungkan: StopContactStreaming menghubungkan: SubmitContactEvaluation

Awalan layanan	Tindakan
	menghubungkan: SuspendContactRecording menghubungkan: TransferContact menghubungkan: UpdateAgentStatus menghubungkan: UpdateAuthenticationProfile menghubungkan: UpdateContact menghubungkan: UpdateContactAttributes menghubungkan: UpdateContactEvaluation menghubungkan: UpdateContactFlowContent menghubungkan: UpdateContactFlowMetadata menghubungkan: UpdateContactFlowModuleContent menghubungkan: UpdateContactFlowModuleMetadata menghubungkan: UpdateContactFlowName menghubungkan: UpdateContactRoutingData menghubungkan: UpdateContactSchedule menghubungkan: UpdateEvaluationForm menghubungkan: UpdateHoursOfOperation menghubungkan: UpdateInstanceAttribute menghubungkan: UpdateInstanceStorageConfig menghubungkan: UpdateParticipantRoleConfig menghubungkan: UpdatePhoneNumber menghubungkan: UpdatePhoneNumberMetadata

Awalan layanan	Tindakan
	menghubungkan: UpdatePredefinedAttribute menghubungkan: UpdatePrompt menghubungkan: UpdateQueueHoursOfOperation menghubungkan: UpdateQueueMaxContacts menghubungkan: UpdateQueueName menghubungkan: UpdateQueueOutboundCallerConfig menghubungkan: UpdateQueueStatus menghubungkan: UpdateQuickConnectConfig menghubungkan: UpdateQuickConnectName menghubungkan: UpdateRoutingProfileAgentAvailabilityTimer menghubungkan: UpdateRoutingProfileConcurrency menghubungkan: UpdateRoutingProfileDefaultOutboundQueue menghubungkan: UpdateRoutingProfileName menghubungkan: UpdateRoutingProfileQueues menghubungkan: UpdateRule menghubungkan: UpdateSecurityProfile menghubungkan: UpdateTaskTemplate menghubungkan: UpdateTrafficDistribution menghubungkan: UpdateUserHierarchy menghubungkan: UpdateUserHierarchyGroupName menghubungkan: UpdateUserHierarchyStructure

Awalan layanan	Tindakan
	menghubungkan: UpdateUserIdentityInfo menghubungkan: UpdateUserPhoneConfig menghubungkan: UpdateUserProficiencies menghubungkan: UpdateUserRoutingProfile menghubungkan: UpdateUserSecurityProfiles menghubungkan: UpdateViewContent menghubungkan: UpdateViewMetadata
cur	kelebar: DeleteReportDefinition kelebar: DescribeReportDefinitions kelebar: ModifyReportDefinition kelebar: PutReportDefinition

Awalan layanan	Tindakan
databrew	database: BatchDeleteRecipeVersion database: CreateDataset database: CreateProfileJob database: CreateProject database: CreateRecipe database: CreateRecipeJob database: CreateRuleset database: CreateSchedule database: DeleteDataset database: DeleteJob database: DeleteProject database: DeleteRecipeVersion database: DeleteRuleset database: DeleteSchedule database: DescribeDataset database: DescribeJob database: DescribeJobRun database: DescribeProject database: DescribeRecipe database: DescribeRuleset database: DescribeSchedule

Awalan layanan	Tindakan
	database: ListDatasets database: ListJobRuns database: ListJobs database: ListProjects database: ListRecipes database: ListRecipeVersions database: ListRulesets database: ListSchedules database: PublishRecipe database: SendProjectSessionAction database: StartJobRun database: StartProjectSession database: StopJobRun database: UpdateDataset database: UpdateProfileJob database: UpdateProject database: UpdateRecipe database: UpdateRecipeJob database: UpdateRuleset database: UpdateSchedule

Awalan layanan	Tindakan
pertukaran data	pertukaran data: CancelJob
	pertukaran data: CreateDataSet
	pertukaran data: CreateEventAction
	pertukaran data: CreateJob
	pertukaran data: CreateRevision
	pertukaran data: DeleteAsset
	pertukaran data: DeleteEventAction
	pertukaran data: DeleteRevision
	pertukaran data: GetEventAction
	pertukaran data: GetJob
	pertukaran data: ListDataSetRevisions
	pertukaran data: ListDataSets
	pertukaran data: ListEventActions
	pertukaran data: ListJobs
	pertukaran data: ListRevisionAssets
	pertukaran data: RevokeRevision
	pertukaran data: SendDataSetNotification
	pertukaran data: StartJob
	pertukaran data: UpdateAsset
	pertukaran data: UpdateDataSet
	pertukaran data: UpdateEventAction

Awalan layanan	Tindakan
	pertukaran data: UpdateRevision
datapipeline	datapipeline: ActivatePipeline datapipeline: CreatePipeline datapipeline: DeactivatePipeline datapipeline: DeletePipeline datapipeline: DescribeObjects datapipeline: DescribePipelines datapipeline: EvaluateExpression datapipeline: GetPipelineDefinition datapipeline: ListPipelines datapipeline: PollForTask datapipeline: PutPipelineDefinition datapipeline: QueryObjects datapipeline: ReportTaskProgress datapipeline: ReportTaskRunnerHeartbeat datapipeline: SetStatus datapipeline: SetTaskStatus datapipeline: ValidatePipelineDefinition

Awalan layanan	Tindakan
dax	dax: CreateCluster
	dax: DecreaseReplicationFactor
	dax: DeleteCluster
	dax: DeleteParameterGroup
	dax: DeleteSubnetGroup
	dax: DescribeClusters
	dax: DescribeDefaultParameters
	dax: DescribeEvents
	dax: DescribeParameterGroups
	dax: DescribeParameters
	dax: DescribeSubnetGroups
	dax: IncreaseReplicationFactor
	dax: RebootNode
	dax: UpdateCluster
	dax: UpdateParameterGroup
	dax: UpdateSubnetGroup

Awalan layanan	Tindakan
devicefarm	devicefarm: CreateDevicePool devicefarm: CreateInstanceProfile devicefarm: CreateNetworkProfile devicefarm: CreateProject devicefarm: CreateRemoteAccessSession devicefarm: CreateTestGridProject devicefarm: CreateTestGridUrl devicefarm: CreateUpload DeviceFarm:C reateVPCEConfiguration devicefarm: DeleteDevicePool devicefarm: DeleteInstanceProfile devicefarm: DeleteNetworkProfile devicefarm: DeleteProject devicefarm: DeleteRemoteAccessSession devicefarm: DeleteRun devicefarm: DeleteTestGridProject devicefarm: DeleteUpload DeviceFarm:D eleteVPCEConfiguration devicefarm: GetAccountSettings devicefarm: GetDevice devicefarm: GetDeviceInstance

Awalan layanan	Tindakan
	devicefarm: GetDevicePool devicefarm: GetDevicePoolCompatibility devicefarm: GetInstanceProfile devicefarm: GetJob devicefarm: GetNetworkProfile devicefarm: GetOfferingStatus devicefarm: GetProject devicefarm: GetRemoteAccessSession devicefarm: GetRun devicefarm: GetSuite devicefarm: GetTest devicefarm: GetTestGridProject devicefarm: GetTestGridSession devicefarm: GetUpload DeviceFarm:G etVPCEConfiguration devicefarm: ListArtifacts devicefarm: ListDeviceInstances devicefarm: ListDevicePools devicefarm: ListDevices devicefarm: ListInstanceProfiles devicefarm: ListJobs

Awalan layanan	Tindakan
	devicefarm: ListNetworkProfiles devicefarm: ListOfferingPromotions devicefarm: ListOfferings devicefarm: ListOfferingTransactions devicefarm: ListProjects devicefarm: ListRemoteAccessSessions devicefarm: ListRuns devicefarm: ListSamples devicefarm: ListSuites devicefarm: ListTestGridProjects devicefarm: ListTestGridSessionActions devicefarm: ListTestGridSessionArtifacts devicefarm: ListTestGridSessions devicefarm: ListTests devicefarm: ListUniqueProblems devicefarm: ListUploads PerangkatFarm:L istVPCEConfigurations devicefarm: PurchaseOffering devicefarm: RenewOffering devicefarm: ScheduleRun devicefarm: StopJob

Awalan layanan	Tindakan
	devicefarm: StopRemoteAccessSession devicefarm: StopRun devicefarm: UpdateDeviceInstance devicefarm: UpdateDevicePool devicefarm: UpdateInstanceProfile devicefarm: UpdateNetworkProfile devicefarm: UpdateProject devicefarm: UpdateTestGridProject devicefarm: UpdateUpload DeviceFarm:U pdateVPCEConfiguration

Awalan layanan	Tindakan
devops-guru	devops-guru: AddNotificationChannel
	devops-guru: DeleteInsight
	devops-guru: DescribeAccountHealth
	devops-guru: DescribeAccountOverview
	devops-guru: DescribeAnomaly
	devops-guru: DescribeEventSourcesConfig
	devops-guru: DescribeFeedback
	devops-guru: DescribeInsight
	devops-guru: DescribeOrganizationHealth
	devops-guru: DescribeOrganizationOverview
	devops-guru: DescribeOrganizationResourceCollectionHealth
	devops-guru: DescribeResourceCollectionHealth
	devops-guru: DescribeServiceIntegration
	devops-guru: GetCostEstimation
	devops-guru: GetResourceCollection
	devops-guru: ListAnomaliesForInsight
	devops-guru: ListAnomalousLogGroups
	devops-guru: ListEvents
	devops-guru: ListInsights
	devops-guru: ListMonitoredResources
	devops-guru: ListNotificationChannels

Awalan layanan	Tindakan
	devops-guru: ListOrganizationInsights
	devops-guru: ListRecommendations
	devops-guru: PutFeedback
	devops-guru: RemoveNotificationChannel
	devops-guru: SearchInsights
	devops-guru: SearchOrganizationInsights
	devops-guru: StartCostEstimation
	devops-guru: UpdateEventSourcesConfig
	devops-guru: UpdateResourceCollection
	devops-guru: UpdateServiceIntegration

Awalan layanan	Tindakan
DirectConnect	directconnect: AcceptDirectConnectGatewayAssociationProposal directconnect: AllocateConnectionOnInterconnect directconnect: AllocateHostedConnection directconnect: AllocatePrivateVirtualInterface directconnect: AllocatePublicVirtualInterface directconnect: AllocateTransitVirtualInterface directconnect: AssociateConnectionWithLag directconnect: AssociateHostedConnection directconnect: AssociateMacSecKey directconnect: AssociateVirtualInterface directconnect: ConfirmConnection directconnect: ConfirmCustomerAgreement directconnect: ConfirmPrivateVirtualInterface directconnect: ConfirmPublicVirtualInterface directconnect: ConfirmTransitVirtualInterface DirectConnect:C reateBGPPeer directconnect: CreateConnection directconnect: CreateDirectConnectGateway directconnect: CreateDirectConnectGatewayAssociation directconnect: CreateDirectConnectGatewayAssociationProposal directconnect: CreateInterconnect

Awalan layanan	Tindakan
	directconnect: CreateLag directconnect: CreatePrivateVirtualInterface directconnect: CreatePublicVirtualInterface directconnect: CreateTransitVirtualInterface DirectConnect:D eleteBGPPeer directconnect: DeleteConnection directconnect: DeleteDirectConnectGateway directconnect: DeleteDirectConnectGatewayAssociation directconnect: DeleteDirectConnectGatewayAssociationProposal directconnect: DeleteInterconnect directconnect: DeleteLag directconnect: DeleteVirtualInterface directconnect: DescribeConnectionLoa directconnect: DescribeConnections directconnect: DescribeConnectionsOnInterconnect directconnect: DescribeCustomerMetadata directconnect: DescribeDirectConnectGatewayAssociationProposals directconnect: DescribeDirectConnectGatewayAssociations directconnect: DescribeDirectConnectGatewayAttachments directconnect: DescribeDirectConnectGateways

Awalan layanan	Tindakan
	directconnect: DescribeHostedConnections directconnect: DescribeInterconnectLoa directconnect: DescribeInterconnects directconnect: DescribeLags directconnect: DescribeLoa directconnect: DescribeLocations directconnect: DescribeRouterConfiguration directconnect: DescribeVirtualGateways directconnect: DescribeVirtualInterfaces directconnect: DisassociateConnectionFromLag directconnect: DisassociateMacSecKey directconnect: ListVirtualInterfaceTestHistory directconnect: StartBgpFailoverTest directconnect: StopBgpFailoverTest directconnect: UpdateConnection directconnect: UpdateDirectConnectGateway directconnect: UpdateDirectConnectGatewayAssociation directconnect: UpdateLag directconnect: UpdateVirtualInterfaceAttributes

Awalan layanan	Tindakan
dlm	dlm: CreateLifecyclePolicy
	dlm: DeleteLifecyclePolicy
	dlm: GetLifecyclePolicies
	dlm: GetLifecyclePolicy
	dlm: UpdateLifecyclePolicy

Awalan layanan	Tindakan
dms	dms: ApplyPendingMaintenanceAction dms: BatchStartRecommendations dms: CancelReplicationTaskAssessmentRun dms: CreateDataProvider dms: CreateEndpoint dms: CreateEventSubscription dms: CreateInstanceProfile dms: CreateMigrationProject dms: CreateReplicationConfig dms: CreateReplicationInstance dms: CreateReplicationSubnetGroup dms: CreateReplicationTask dms: DeleteCertificate dms: DeleteConnection dms: DeleteDataProvider dms: DeleteEndpoint dms: DeleteEventSubscription dms: DeleteFleetAdvisorCollector dms: DeleteFleetAdvisorDatabases dms: DeleteInstanceProfile dms: DeleteMigrationProject

Awalan layanan	Tindakan
	dms: DeleteReplicationConfig dms: DeleteReplicationInstance dms: DeleteReplicationSubnetGroup dms: DeleteReplicationTask dms: DeleteReplicationTaskAssessmentRun dms: DescribeAccountAttributes dms: DescribeApplicableIndividualAssessments dms: DescribeCertificates dms: DescribeConnections dms: DescribeEndpoints dms: DescribeEndpointSettings dms: DescribeEndpointTypes dms: DescribeEngineVersions dms: DescribeEventCategories dms: DescribeEvents dms: DescribeEventSubscriptions dms: DescribeFleetAdvisorCollectors dms: DescribeFleetAdvisorDatabases dms: DescribeFleetAdvisorLsaAnalysis dms: DescribeFleetAdvisorSchemaObjectSummary dms: DescribeFleetAdvisorSchemas

Awalan layanan	Tindakan
	dms: DescribeMetadataModelImports dms: DescribeOrderableReplicationInstances dms: DescribePendingMaintenanceActions dms: DescribeRecommendationLimitations dms: DescribeRecommendations dms: DescribeRefreshSchemasStatus dms: DescribeReplicationConfigs dms: DescribeReplicationInstances dms: DescribeReplicationInstanceTaskLogs dms: DescribeReplications dms: DescribeReplicationSubnetGroups dms: DescribeReplicationTableStatistics dms: DescribeReplicationTaskAssessmentResults dms: DescribeReplicationTaskAssessmentRuns dms: DescribeReplicationTaskIndividualAssessments dms: DescribeReplicationTasks dms: DescribeSchemas dms: DescribeTableStatistics dms: ExportMetadataModelAssessment dms: GetMetadataModel dms: ImportCertificate

Awalan layanan	Tindakan
	dms: ListMetadataModelAssessmentActionItems
	dms: ModifyEndpoint
	dms: ModifyEventSubscription
	dms: ModifyReplicationConfig
	dms: ModifyReplicationInstance
	dms: ModifyReplicationSubnetGroup
	dms: ModifyReplicationTask
	dms: MoveReplicationTask
	dms: RebootReplicationInstance
	dms: RefreshSchemas
	dms: ReloadReplicationTables
	dms: ReloadTables
	dms: RunFleetAdvisorLsaAnalysis
	dms: StartMetadataModelAssessment
	dms: StartMetadataModelConversion
	dms: StartMetadataModelExportToTarget
	dms: StartRecommendations
	dms: StartReplication
	dms: StartReplicationTask
	dms: StartReplicationTaskAssessment
	dms: StopReplicationTask

Awalan layanan	Tindakan
	dms: TestConnection dms: UpdateSubscriptionsToEventBridge
docdb-elastis	docdb-elastis: CopyClusterSnapshot docdb-elastis: DeleteCluster docdb-elastis: DeleteClusterSnapshot docdb-elastis: GetCluster docdb-elastis: GetClusterSnapshot docdb-elastis: ListClusters docdb-elastis: ListClusterSnapshots docdb-elastis: RestoreClusterFromSnapshot docdb-elastis: StartCluster docdb-elastis: StopCluster docdb-elastis: UpdateCluster

Awalan layanan	Tindakan
dynamodb	dinamodb: CreateBackup dinamodb: CreateGlobalTable dinamodb: CreateTable dinamodb: DeleteBackup dinamodb: DeleteTable dinamodb: DescribeBackup dinamodb: DescribeContinuousBackups dinamodb: DescribeContributorInsights dinamodb: DescribeEndpoints dinamodb: DescribeExport dinamodb: DescribeGlobalTable dinamodb: DescribeGlobalTableSettings dinamodb: DescribeImport dinamodb: DescribeKinesisStreamingDestination dinamodb: DescribeLimits dinamodb: DescribeStream dinamodb: DescribeTable dinamodb: DescribeTableReplicaAutoScaling dinamodb: DescribeTimeToLive dinamodb: DisableKinesisStreamingDestination dinamodb: EnableKinesisStreamingDestination

Awalan layanan	Tindakan
	dinamodb: ExportTableToPointInTime dinamodb: GetResourcePolicy dinamodb: ImportTable dinamodb: ListBackups dinamodb: ListContributorInsights dinamodb: ListExports dinamodb: ListGlobalTables dinamodb: ListImports dinamodb: ListStreams dinamodb: ListTables dinamodb: RestoreTableFromBackup dinamodb: RestoreTableToPointInTime dinamodb: UpdateContinuousBackups dinamodb: UpdateContributorInsights dinamodb: UpdateGlobalTable dinamodb: UpdateGlobalTableSettings dinamodb: UpdateKinesisStreamingDestination dinamodb: UpdateTable dinamodb: UpdateTableReplicaAutoScaling dinamodb: UpdateTimeToLive

Awalan layanan	Tindakan
ebs	ebs: CompleteSnapshot ebs: StartSnapshot

Awalan layanan	Tindakan
ec2	EC2: AcceptAddressTransfer EC2: AcceptReservedInstancesExchangeQuote EC2: AcceptTransitGatewayMulticastDomainAssociations EC2: AcceptTransitGatewayPeeringAttachment EC2: AcceptTransitGatewayVpcAttachment EC2: AcceptVpcEndpointConnections EC2: AcceptVpcPeeringConnection EC2: AdvertiseByoipCidr EC2: AllocateAddress EC2: AllocateHosts EC2: AllocateIpamPoolCidr EC2: ApplySecurityGroupsToClientVpnTargetNetwork ec2:6Alamat AssignIpv EC2: AssignPrivateIpAddresses EC2: AssignPrivateNatGatewayAddress EC2: AssociateAddress EC2: AssociateClientVpnTargetNetwork EC2: AssociateDhcpOptions EC2: AssociateEnclaveCertificateIamRole EC2: AssociateIamInstanceProfile EC2: AssociateInstanceEventWindow

Awalan layanan	Tindakan
	EC2: AssociateIpamByoasn EC2: AssociateIpamResourceDiscovery EC2: AssociateNatGatewayAddress EC2: AssociateRouteTable EC2: AssociateSubnetCidrBlock EC2: AssociateTransitGatewayMulticastDomain EC2: AssociateTransitGatewayPolicyTable EC2: AssociateTransitGatewayRouteTable EC2: AssociateTrunkInterface EC2: AssociateVpcCidrBlock EC2: AttachClassicLinkVpc EC2: AttachInternetGateway EC2: AttachNetworkInterface EC2: AttachVerifiedAccessTrustProvider EC2: AttachVolume EC2: AttachVpnGateway EC2: AuthorizeClientVpnIngress EC2: AuthorizeSecurityGroupEgress EC2: AuthorizeSecurityGroupIngress EC2: BundleInstance EC2: CancelBundleTask

Awalan layanan	Tindakan
	EC2: CancelCapacityReservation EC2: CancelCapacityReservationFleets EC2: CancelConversionTask EC2: CancelExportTask EC2: CancelImageLaunchPermission EC2: CancelImportTask EC2: CancelReservedInstancesListing EC2: CancelSpotFleetRequests EC2: CancelSpotInstanceRequests EC2: ConfirmProductInstance EC2: CopyFpgaImage EC2: CopyImage EC2: CopySnapshot EC2: CreateCapacityReservation EC2: CreateCapacityReservationBySplitting EC2: CreateCapacityReservationFleet EC2: CreateCarrierGateway EC2: CreateClientVpnEndpoint EC2: CreateClientVpnRoute EC2: CreateCoipCidr EC2: CreateCoipPool

Awalan layanan	Tindakan
	EC2: CreateCustomerGateway EC2: CreateDefaultSubnet EC2: CreateDefaultVpc EC2: CreateDhcpOptions EC2: CreateEgressOnlyInternetGateway EC2: CreateFleet EC2: CreateFlowLogs EC2: CreateFpgaImage EC2: CreateImage EC2: CreateInstanceConnectEndpoint EC2: CreateInstanceEventWindow EC2: CreateInstanceExportTask EC2: CreateInternetGateway EC2: CreateIpam EC2: CreateIpamExternalResourceVerificationToken EC2: CreateIpamPool EC2: CreateIpamResourceDiscovery EC2: CreateIpamScope EC2: CreateKeyPair EC2: CreateLaunchTemplateVersion EC2: CreateLocalGatewayRoute

Awalan layanan	Tindakan
	EC2: CreateLocalGatewayRouteTable EC2: CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation EC2: CreateLocalGatewayRouteTableVpcAssociation EC2: CreateManagedPrefixList EC2: CreateNatGateway EC2: CreateNetworkAcl EC2: CreateNetworkAclEntry EC2: CreateNetworkInsightsAccessScope EC2: CreateNetworkInsightsPath EC2: CreateNetworkInterface EC2: CreateNetworkInterfacePermission EC2: CreatePlacementGroup ec2:4Kolam CreatePublicIpv EC2: CreateReplaceRootVolumeTask EC2: CreateReservedInstancesListing EC2: CreateRestoreImageTask EC2: CreateRoute EC2: CreateRouteTable EC2: CreateSecurityGroup EC2: CreateSnapshots

Awalan layanan	Tindakan
	EC2: CreateSpotDatafeedSubscription
	EC2: CreateStoreImageTask
	EC2: CreateSubnet
	EC2: CreateSubnetCidrReservation
	EC2: CreateTrafficMirrorFilter
	EC2: CreateTrafficMirrorFilterRule
	EC2: CreateTrafficMirrorSession
	EC2: CreateTrafficMirrorTarget
	EC2: CreateTransitGateway
	EC2: CreateTransitGatewayConnect
	EC2: CreateTransitGatewayConnectPeer
	EC2: CreateTransitGatewayMulticastDomain
	EC2: CreateTransitGatewayPeeringAttachment
	EC2: CreateTransitGatewayPolicyTable
	EC2: CreateTransitGatewayPrefixListReference
	EC2: CreateTransitGatewayRoute
	EC2: CreateTransitGatewayRouteTable
	EC2: CreateTransitGatewayRouteTableAnnouncement
	EC2: CreateTransitGatewayVpcAttachment
	EC2: CreateVerifiedAccessEndpoint
	EC2: CreateVerifiedAccessGroup

Awalan layanan	Tindakan
	EC2: CreateVerifiedAccessInstance
	EC2: CreateVerifiedAccessTrustProvider
	EC2: CreateVolume
	EC2: CreateVpc
	EC2: CreateVpcEndpoint
	EC2: CreateVpcEndpointConnectionNotification
	EC2: CreateVpcEndpointServiceConfiguration
	EC2: CreateVpcPeeringConnection
	EC2: CreateVpnConnection
	EC2: CreateVpnConnectionRoute
	EC2: CreateVpnGateway
	EC2: DeleteCarrierGateway
	EC2: DeleteClientVpnEndpoint
	EC2: DeleteClientVpnRoute
	EC2: DeleteCoipCidr
	EC2: DeleteCoipPool
	EC2: DeleteCustomerGateway
	EC2: DeleteDhcpOptions
	EC2: DeleteEgressOnlyInternetGateway
	EC2: DeleteFleets
	EC2: DeleteFlowLogs

Awalan layanan	Tindakan
	EC2: DeleteFpgaImage EC2: DeleteInstanceConnectEndpoint EC2: DeleteInstanceEventWindow EC2: DeleteInternetGateway EC2: DeleteIpam EC2: DeleteIpamExternalResourceVerificationToken EC2: DeleteIpamPool EC2: DeleteIpamResourceDiscovery EC2: DeleteIpamScope EC2: DeleteKeyPair EC2: DeleteLaunchTemplate EC2: DeleteLaunchTemplateVersions EC2: DeleteLocalGatewayRoute EC2: DeleteLocalGatewayRouteTable EC2: DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation EC2: DeleteLocalGatewayRouteTableVpcAssociation EC2: DeleteManagedPrefixList EC2: DeleteNatGateway EC2: DeleteNetworkAcl EC2: DeleteNetworkAclEntry

Awalan layanan	Tindakan
	EC2: DeleteNetworkInsightsAccessScope
	EC2: DeleteNetworkInsightsAccessScopeAnalysis
	EC2: DeleteNetworkInsightsAnalysis
	EC2: DeleteNetworkInsightsPath
	EC2: DeleteNetworkInterface
	EC2: DeleteNetworkInterfacePermission
	EC2: DeletePlacementGroup
	ec2:4Kolam DeletePublicIpv
	EC2: DeleteQueuedReservedInstances
	EC2: DeleteRoute
	EC2: DeleteRouteTable
	EC2: DeleteSecurityGroup
	EC2: DeleteSpotDatafeedSubscription
	EC2: DeleteSubnet
	EC2: DeleteSubnetCidrReservation
	EC2: DeleteTrafficMirrorFilter
	EC2: DeleteTrafficMirrorFilterRule
	EC2: DeleteTrafficMirrorSession
	EC2: DeleteTrafficMirrorTarget
	EC2: DeleteTransitGateway
	EC2: DeleteTransitGatewayConnect

Awalan layanan	Tindakan
	EC2: DeleteTransitGatewayConnectPeer
	EC2: DeleteTransitGatewayMulticastDomain
	EC2: DeleteTransitGatewayPeeringAttachment
	EC2: DeleteTransitGatewayPolicyTable
	EC2: DeleteTransitGatewayPrefixListReference
	EC2: DeleteTransitGatewayRoute
	EC2: DeleteTransitGatewayRouteTable
	EC2: DeleteTransitGatewayRouteTableAnnouncement
	EC2: DeleteTransitGatewayVpcAttachment
	EC2: DeleteVerifiedAccessEndpoint
	EC2: DeleteVerifiedAccessGroup
	EC2: DeleteVerifiedAccessInstance
	EC2: DeleteVerifiedAccessTrustProvider
	EC2: DeleteVolume
	EC2: DeleteVpc
	EC2: DeleteVpcEndpointConnectionNotifications
	EC2: DeleteVpcEndpoints
	EC2: DeleteVpcEndpointServiceConfigurations
	EC2: DeleteVpcPeeringConnection
	EC2: DeleteVpnConnection
	EC2: DeleteVpnConnectionRoute

Awalan layanan	Tindakan
	EC2: DeleteVpnGateway EC2: DeprovisionByoipCidr EC2: DeprovisionIpamByoasn EC2: DeprovisionIpamPoolCidr ec2: DeprovisionPublicIpv 4 PoolCidr EC2: DeregisterImage EC2: DeregisterInstanceEventNotificationAttributes EC2: DeregisterTransitGatewayMulticastGroupMembers EC2: DeregisterTransitGatewayMulticastGroupSources EC2: DescribeAccountAttributes EC2: DescribeAddresses EC2: DescribeAddressesAttribute EC2: DescribeAddressTransfers EC2: DescribeAggregateIdFormat EC2: DescribeAvailabilityZones EC2: DescribeAwsNetworkPerformanceMetricSubscriptions EC2: DescribeBundleTasks EC2: DescribeByoipCidrs EC2: DescribeCapacityReservationFleets EC2: DescribeCapacityReservations EC2: DescribeCarrierGateways

Awalan layanan	Tindakan
	EC2: DescribeClassicLinkInstances
	EC2: DescribeClientVpnAuthorizationRules
	EC2: DescribeClientVpnConnections
	EC2: DescribeClientVpnEndpoints
	EC2: DescribeClientVpnRoutes
	EC2: DescribeClientVpnTargetNetworks
	EC2: DescribeCoipPools
	EC2: DescribeConversionTasks
	EC2: DescribeCustomerGateways
	EC2: DescribeDhcpOptions
	EC2: DescribeEgressOnlyInternetGateways
	EC2: DescribeElasticGpus
	EC2: DescribeExportImageTasks
	EC2: DescribeExportTasks
	EC2: DescribeFastLaunchImages
	EC2: DescribeFastSnapshotRestores
	EC2: DescribeFleetHistory
	EC2: DescribeFleetInstances
	EC2: DescribeFleets
	EC2: DescribeFlowLogs
	EC2: DescribeFpgaImageAttribute

Awalan layanan	Tindakan
	EC2: DescribeFpgaImages EC2: DescribeHostReservationOfferings EC2: DescribeHostReservations EC2: DescribeHosts EC2: DescribeIamInstanceProfileAssociations EC2: DescribeIdentityIdFormat EC2: DescribeIdFormat EC2: DescribeImageAttribute EC2: DescribeImportImageTasks EC2: DescribeImportSnapshotTasks EC2: DescribeInstanceConnectEndpoints EC2: DescribeInstanceCreditSpecifications EC2: DescribeInstanceEventNotificationAttributes EC2: DescribeInstanceEventWindows EC2: DescribeInstanceTopology EC2: DescribeInstanceTypes EC2: DescribeInternetGateways EC2: DescribeIpamByoasn EC2: DescribeIpamExternalResourceVerificationTokens EC2: DescribeIpamPools EC2: DescribeIpamResourceDiscoveries

Awalan layanan	Tindakan
	EC2: DescribeIpamResourceDiscoveryAssociations EC2: DescribeIpams EC2: DescribeIpamScopes ec2:6 Kolam DescribeIpv EC2: DescribeKeyPairs EC2: DescribeLocalGatewayRouteTables EC2: DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations EC2: DescribeLocalGatewayRouteTableVpcAssociations EC2: DescribeLocalGateways EC2: DescribeLocalGatewayVirtualInterfaceGroups EC2: DescribeLocalGatewayVirtualInterfaces EC2: DescribeLockedSnapshots EC2: DescribeMacHosts EC2: DescribeManagedPrefixLists EC2: DescribeMovingAddresses EC2: DescribeNatGateways EC2: DescribeNetworkAcls EC2: DescribeNetworkInsightsAccessScopeAnalyses EC2: DescribeNetworkInsightsAccessScopes EC2: DescribeNetworkInsightsAnalyses

Awalan layanan	Tindakan
	EC2: DescribeNetworkInsightsPaths EC2: DescribeNetworkInterfaceAttribute EC2: DescribeNetworkInterfacePermissions EC2: DescribeNetworkInterfaces EC2: DescribePlacementGroups EC2: DescribePrefixLists EC2: DescribePrincipalIdFormat ec2:4Pools DescribePublicIpv EC2: DescribeRegions EC2: DescribeReplaceRootVolumeTasks EC2: DescribeReservedInstances EC2: DescribeReservedInstancesListings EC2: DescribeReservedInstancesModifications EC2: DescribeReservedInstancesOfferings EC2: DescribeRouteTables EC2: DescribeScheduledInstanceAvailability EC2: DescribeScheduledInstances EC2: DescribeSecurityGroupReferences EC2: DescribeSecurityGroupRules EC2: DescribeSecurityGroups EC2: DescribeSnapshotAttribute

Awalan layanan	Tindakan
	EC2: DescribeSnapshotTierStatus
	EC2: DescribeSpotDatafeedSubscription
	EC2: DescribeSpotFleetInstances
	EC2: DescribeSpotFleetRequestHistory
	EC2: DescribeSpotFleetRequests
	EC2: DescribeSpotInstanceRequests
	EC2: DescribeSpotPriceHistory
	EC2: DescribeStaleSecurityGroups
	EC2: DescribeStoreImageTasks
	EC2: DescribeTrafficMirrorFilterRules
	EC2: DescribeTrafficMirrorFilters
	EC2: DescribeTrafficMirrorSessions
	EC2: DescribeTrafficMirrorTargets
	EC2: DescribeTransitGatewayAttachments
	EC2: DescribeTransitGatewayConnectPeers
	EC2: DescribeTransitGatewayConnects
	EC2: DescribeTransitGatewayMulticastDomains
	EC2: DescribeTransitGatewayPeeringAttachments
	EC2: DescribeTransitGatewayPolicyTables
	EC2: DescribeTransitGatewayRouteTableAnnouncements
	EC2: DescribeTransitGatewayRouteTables

Awalan layanan	Tindakan
	EC2: DescribeTransitGateways
	EC2: DescribeTransitGatewayVpcAttachments
	EC2: DescribeTrunkInterfaceAssociations
	EC2: DescribeVerifiedAccessEndpoints
	EC2: DescribeVerifiedAccessGroups
	EC2: DescribeVerifiedAccessInstanceLoggingConfigurations
	EC2: DescribeVerifiedAccessInstances
	EC2: DescribeVerifiedAccessTrustProviders
	EC2: DescribeVolumeAttribute
	EC2: DescribeVolumes
	EC2: DescribeVolumesModifications
	EC2: DescribeVolumeStatus
	EC2: DescribeVpcAttribute
	EC2: DescribeVpcClassicLink
	EC2: DescribeVpcClassicLinkDnsSupport
	EC2: DescribeVpcEndpointConnectionNotifications
	EC2: DescribeVpcEndpointConnections
	EC2: DescribeVpcEndpoints
	EC2: DescribeVpcEndpointServiceConfigurations
	EC2: DescribeVpcEndpointServicePermissions
	EC2: DescribeVpcEndpointServices

Awalan layanan	Tindakan
	EC2: DescribeVpcPeeringConnections EC2: DescribeVpcs EC2: DescribeVpnConnections EC2: DescribeVpnGateways EC2: DetachClassicLinkVpc EC2: DetachInternetGateway EC2: DetachNetworkInterface EC2: DetachVerifiedAccessTrustProvider EC2: DetachVolume EC2: DetachVpnGateway EC2: DisableAddressTransfer EC2: DisableAwsNetworkPerformanceMetricSubscription EC2: DisableEbsEncryptionByDefault EC2: DisableFastLaunch EC2: DisableFastSnapshotRestores EC2: DisableImage EC2: DisableImageBlockPublicAccess EC2: DisableImageDeprecation EC2: DisableImageDeregistrationProtection EC2: DisableIpamOrganizationAdminAccount EC2: DisableSerialConsoleAccess

Awalan layanan	Tindakan
	EC2: DisableSnapshotBlockPublicAccess
	EC2: DisableTransitGatewayRouteTablePropagation
	EC2: DisableVgwRoutePropagation
	EC2: DisableVpcClassicLink
	EC2: DisableVpcClassicLinkDnsSupport
	EC2: DisassociateAddress
	EC2: DisassociateClientVpnTargetNetwork
	EC2: DisassociateEnclaveCertificateIamRole
	EC2: DisassociateIamInstanceProfile
	EC2: DisassociateInstanceEventWindow
	EC2: DisassociateIpamByoasn
	EC2: DisassociateIpamResourceDiscovery
	EC2: DisassociateNatGatewayAddress
	EC2: DisassociateRouteTable
	EC2: DisassociateSubnetCidrBlock
	EC2: DisassociateTransitGatewayMulticastDomain
	EC2: DisassociateTransitGatewayPolicyTable
	EC2: DisassociateTransitGatewayRouteTable
	EC2: DisassociateTrunkInterface
	EC2: DisassociateVpcCidrBlock
	EC2: EnableAddressTransfer

Awalan layanan	Tindakan
	EC2: EnableAwsNetworkPerformanceMetricSubscription
	EC2: EnableEbsEncryptionByDefault
	EC2: EnableFastLaunch
	EC2: EnableFastSnapshotRestores
	EC2: EnableImage
	EC2: EnableImageBlockPublicAccess
	EC2: EnableImageDeprecation
	EC2: EnableImageDeregistrationProtection
	EC2: EnableIpamOrganizationAdminAccount
	EC2: EnableReachabilityAnalyzerOrganizationSharing
	EC2: EnableSerialConsoleAccess
	EC2: EnableSnapshotBlockPublicAccess
	EC2: EnableTransitGatewayRouteTablePropagation
	EC2: EnableVgwRoutePropagation
	ec2: EnableVolume IO
	EC2: EnableVpcClassicLink
	EC2: EnableVpcClassicLinkDnsSupport
	EC2: ExportClientVpnClientCertificateRevocationList
	EC2: ExportClientVpnClientConfiguration
	EC2: ExportImage
	EC2: ExportTransitGatewayRoutes

Awalan layanan	Tindakan
	EC2: GetAssociatedEnclaveCertificateIamRoles
	ec2:6 GetAssociatedIpv PoolCidrs
	EC2: GetAwsNetworkPerformanceData
	EC2: GetCapacityReservationUsage
	EC2: GetCoipPoolUsage
	EC2: GetConsoleOutput
	EC2: GetConsoleScreenshot
	EC2: GetDefaultCreditSpecification
	EC2: GetEbsDefaultKmsKeyId
	EC2: GetEbsEncryptionByDefault
	EC2: GetFlowLogsIntegrationTemplate
	EC2: GetGroupsForCapacityReservation
	EC2: GetHostReservationPurchasePreview
	EC2: GetImageBlockPublicAccessState
	EC2: GetInstanceMetadataDefaults
	EC2: GetInstanceTpmEkPub
	EC2: GetInstanceTypesFromInstanceRequirements
	EC2: GetInstanceUefiData
	EC2: GetIpamAddressHistory
	EC2: GetIpamDiscoveredAccounts
	EC2: GetIpamDiscoveredPublicAddresses

Awalan layanan	Tindakan
	EC2: GetIpamDiscoveredResourceCidrs
	EC2: GetIpamPoolAllocations
	EC2: GetIpamPoolCidrs
	EC2: GetIpamResourceCidrs
	EC2: GetLaunchTemplateData
	EC2: GetManagedPrefixListAssociations
	EC2: GetManagedPrefixListEntries
	EC2: GetNetworkInsightsAccessScopeAnalysisFindings
	EC2: GetNetworkInsightsAccessScopeContent
	EC2: GetPasswordData
	EC2: GetReservedInstancesExchangeQuote
	EC2: GetSecurityGroupsForVpc
	EC2: GetSerialConsoleAccessStatus
	EC2: GetSnapshotBlockPublicAccessState
	EC2: GetSpotPlacementScores
	EC2: GetSubnetCidrReservations
	EC2: GetTransitGatewayAttachmentPropagations
	EC2: GetTransitGatewayMulticastDomainAssociations
	EC2: GetTransitGatewayPolicyTableAssociations
	EC2: GetTransitGatewayPolicyTableEntries
	EC2: GetTransitGatewayPrefixListReferences

Awalan layanan	Tindakan
	EC2: GetTransitGatewayRouteTableAssociations
	EC2: GetTransitGatewayRouteTablePropagations
	EC2: GetVerifiedAccessEndpointPolicy
	EC2: GetVerifiedAccessGroupPolicy
	EC2: GetVpnConnectionDeviceSampleConfiguration
	EC2: GetVpnConnectionDeviceTypes
	EC2: GetVpnTunnelReplacementStatus
	EC2: ImportClientVpnClientCertificateRevocationList
	EC2: ImportImage
	EC2: ImportInstance
	EC2: ImportKeyPair
	EC2: ImportSnapshot
	EC2: ImportVolume
	EC2: ListImagesInRecycleBin
	EC2: ListSnapshotsInRecycleBin
	EC2: LockSnapshot
	EC2: ModifyAddressAttribute
	EC2: ModifyAvailabilityZoneGroup
	EC2: ModifyCapacityReservation
	EC2: ModifyCapacityReservationFleet
	EC2: ModifyClientVpnEndpoint

Awalan layanan	Tindakan
	EC2: ModifyDefaultCreditSpecification
	EC2: ModifyEbsDefaultKmsKeyId
	EC2: ModifyFleet
	EC2: ModifyFpgaImageAttribute
	EC2: ModifyHosts
	EC2: ModifyIdentityIdFormat
	EC2: ModifyIdFormat
	EC2: ModifyImageAttribute
	EC2: ModifyInstanceAttribute
	EC2: ModifyInstanceCapacityReservationAttributes
	EC2: ModifyInstanceCreditSpecification
	EC2: ModifyInstanceEventStartTime
	EC2: ModifyInstanceEventWindow
	EC2: ModifyInstanceMaintenanceOptions
	EC2: ModifyInstanceMetadataDefaults
	EC2: ModifyInstanceMetadataOptions
	EC2: ModifyInstancePlacement
	EC2: ModifyIpam
	EC2: ModifyIpamPool
	EC2: ModifyIpamResourceCidr
	EC2: ModifyIpamResourceDiscovery

Awalan layanan	Tindakan
	EC2: ModifyIpamScope
	EC2: ModifyLaunchTemplate
	EC2: ModifyLocalGatewayRoute
	EC2: ModifyManagedPrefixList
	EC2: ModifyNetworkInterfaceAttribute
	EC2: ModifyPrivateDnsNameOptions
	EC2: ModifyReservedInstances
	EC2: ModifySecurityGroupRules
	EC2: ModifySnapshotAttribute
	EC2: ModifySnapshotTier
	EC2: ModifySpotFleetRequest
	EC2: ModifySubnetAttribute
	EC2: ModifyTrafficMirrorFilterNetworkServices
	EC2: ModifyTrafficMirrorFilterRule
	EC2: ModifyTrafficMirrorSession
	EC2: ModifyTransitGateway
	EC2: ModifyTransitGatewayPrefixListReference
	EC2: ModifyTransitGatewayVpcAttachment
	EC2: ModifyVerifiedAccessEndpoint
	EC2: ModifyVerifiedAccessEndpointPolicy
	EC2: ModifyVerifiedAccessGroup

Awalan layanan	Tindakan
	EC2: ModifyVerifiedAccessGroupPolicy
	EC2: ModifyVerifiedAccessInstance
	EC2: ModifyVerifiedAccessInstanceLoggingConfiguration
	EC2: ModifyVerifiedAccessTrustProvider
	EC2: ModifyVolume
	EC2: ModifyVolumeAttribute
	EC2: ModifyVpcAttribute
	EC2: ModifyVpcEndpoint
	EC2: ModifyVpcEndpointConnectionNotification
	EC2: ModifyVpcEndpointServiceConfiguration
	EC2: ModifyVpcEndpointServicePayerResponsibility
	EC2: ModifyVpcEndpointServicePermissions
	EC2: ModifyVpcPeeringConnectionOptions
	EC2: ModifyVpcTenancy
	EC2: ModifyVpnConnection
	EC2: ModifyVpnConnectionOptions
	EC2: ModifyVpnTunnelCertificate
	EC2: ModifyVpnTunnelOptions
	EC2: MonitorInstances
	EC2: MoveAddressToVpc
	EC2: MoveByoipCidrToIpam

Awalan layanan	Tindakan
	EC2: MoveCapacityReservationInstances
	EC2: ProvisionByoipCidr
	EC2: ProvisionIpamByoasn
	EC2: ProvisionIpamPoolCidr
	ec2: ProvisionPublicIpv 4 PoolCidr
	EC2: PurchaseHostReservation
	EC2: PurchaseReservedInstancesOffering
	EC2: PurchaseScheduledInstances
	EC2: RebootInstances
	EC2: RegisterImage
	EC2: RegisterInstanceEventNotificationAttributes
	EC2: RegisterTransitGatewayMulticastGroupMembers
	EC2: RegisterTransitGatewayMulticastGroupSources
	EC2: RejectTransitGatewayMulticastDomainAssociations
	EC2: RejectTransitGatewayPeeringAttachment
	EC2: RejectTransitGatewayVpcAttachment
	EC2: RejectVpcEndpointConnections
	EC2: RejectVpcPeeringConnection
	EC2: ReleaseAddress
	EC2: ReleaseHosts
	EC2: ReleaseIpamPoolAllocation

Awalan layanan	Tindakan
	EC2: ReplaceIamInstanceProfileAssociation
	EC2: ReplaceNetworkAclAssociation
	EC2: ReplaceNetworkAclEntry
	EC2: ReplaceRoute
	EC2: ReplaceRouteTableAssociation
	EC2: ReplaceTransitGatewayRoute
	EC2: ReplaceVpnTunnel
	EC2: ReportInstanceStatus
	EC2: RequestSpotFleet
	EC2: RequestSpotInstances
	EC2: ResetAddressAttribute
	EC2: ResetEbsDefaultKmsKeyId
	EC2: ResetFpgaImageAttribute
	EC2: ResetImageAttribute
	EC2: ResetInstanceAttribute
	EC2: ResetNetworkInterfaceAttribute
	EC2: ResetSnapshotAttribute
	EC2: RestoreAddressToClassic
	EC2: RestoreImageFromRecycleBin
	EC2: RestoreManagedPrefixListVersion
	EC2: RestoreSnapshotFromRecycleBin

Awalan layanan	Tindakan
	EC2: RestoreSnapshotTier
	EC2: RevokeClientVpnIngress
	EC2: RevokeSecurityGroupEgress
	EC2: RevokeSecurityGroupIngress
	EC2: RunInstances
	EC2: RunScheduledInstances
	EC2: SearchLocalGatewayRoutes
	EC2: SearchTransitGatewayMulticastGroups
	EC2: SearchTransitGatewayRoutes
	EC2: SendDiagnosticInterrupt
	EC2: StartInstances
	EC2: StartNetworkInsightsAccessScopeAnalysis
	EC2: StartNetworkInsightsAnalysis
	EC2: StartVpcEndpointServicePrivateDnsVerification
	EC2: TerminateClientVpnConnections
	ec2:6Alamat UnassignIpv
	EC2: UnassignPrivateIpAddresses
	EC2: UnassignPrivateNatGatewayAddress
	EC2: UnlockSnapshot
	EC2: UnmonitorInstances
	EC2: UpdateSecurityGroupRuleDescriptionsEgress

Awalan layanan	Tindakan
	EC2: UpdateSecurityGroupRuleDescriptionsIngress EC2: WithdrawByoipCidr

Awalan layanan	Tindakan
ecr	ecr: BatchCheckLayerAvailability
	ecr: BatchDeleteImage
	ecr: BatchGetImage
	ecr: BatchGetRepositoryScanningConfiguration
	ecr: CompleteLayerUpload
	ecr: CreatePullThroughCacheRule
	ecr: CreateRepositoryCreationTemplate
	ecr: DeleteLifecyclePolicy
	ecr: DeletePullThroughCacheRule
	ecr: DeleteRegistryPolicy
	ecr: DeleteRepository
	ecr: DeleteRepositoryCreationTemplate
	ecr: DeleteRepositoryPolicy
	ecr: DescribeImageReplicationStatus
	ecr: DescribeImages
	ecr: DescribeImageScanFindings
	ecr: DescribePullThroughCacheRules
	ecr: DescribeRegistry
	ecr: DescribeRepositories
	ecr: DescribeRepositoryCreationTemplates
	ecr: GetAccountSetting

Awalan layanan	Tindakan
	ecr: GetAuthorizationToken
	ecr: GetDownloadUrlForLayer
	ecr: GetLifecyclePolicy
	ecr: GetLifecyclePolicyPreview
	ecr: GetRegistryPolicy
	ecr: GetRegistryScanningConfiguration
	ecr: GetRepositoryPolicy
	ecr: InitiateLayerUpload
	ecr: ListImages
	ecr: PutAccountSetting
	ecr: PutImage
	ecr: PutImageScanningConfiguration
	ecr: PutRegistryPolicy
	ecr: PutRegistryScanningConfiguration
	ecr: PutReplicationConfiguration
	ecr: StartImageScan
	ecr: StartLifecyclePolicyPreview
	ecr: UpdatePullThroughCacheRule
	ecr: UpdateRepositoryCreationTemplate
	ecr: UploadLayerPart
	ecr: ValidatePullThroughCacheRule

Awalan layanan	Tindakan
ecr-publik	ecr-publik: BatchCheckLayerAvailability ecr-publik: BatchDeleteImage ecr-publik: CompleteLayerUpload ecr-publik: CreateRepository ecr-publik: DeleteRepository ecr-publik: DeleteRepositoryPolicy ecr-publik: DescribeImages ecr-publik: DescribeRegistries ecr-publik: DescribeRepositories ecr-publik: GetAuthorizationToken ecr-publik: GetRegistryCatalogData ecr-publik: GetRepositoryCatalogData ecr-publik: GetRepositoryPolicy ecr-publik: InitiateLayerUpload ecr-publik: PutImage ecr-publik: PutRegistryCatalogData ecr-publik: PutRepositoryCatalogData ecr-publik: SetRepositoryPolicy ecr-publik: UploadLayerPart

Awalan layanan	Tindakan
ecs	ecs: CreateCapacityProvider
	ecs: CreateCluster
	ecs: CreateService
	ecs: CreateTaskSet
	ecs: DeleteAccountSetting
	ecs: DeleteAttributes
	ecs: DeleteCapacityProvider
	ecs: DeleteCluster
	ecs: DeleteService
	ecs: DeleteTaskDefinitions
	ecs: DeleteTaskSet
	ecs: DeregisterContainerInstance
	ecs: DeregisterTaskDefinition
	ecs: DescribeCapacityProviders
	ecs: DescribeClusters
	ecs: DescribeContainerInstances
	ecs: DescribeServices
	ecs: DescribeTaskDefinition
	ecs: DescribeTasks
	ecs: DescribeTaskSets
	ecs: DiscoverPollEndpoint

Awalan layanan	Tindakan
	ecs: ExecuteCommand
	ecs: GetTaskProtection
	ecs: ListAccountSettings
	ecs: ListAttributes
	ecs: ListClusters
	ecs: ListContainerInstances
	ecs: ListServices
	ecs: ListServicesByNamespace
	ecs: ListTaskDefinitionFamilies
	ecs: ListTaskDefinitions
	ecs: ListTasks
	ecs: PutAccountSetting
	ecs: PutAccountSettingDefault
	ecs: PutAttributes
	ecs: PutClusterCapacityProviders
	ecs: RegisterContainerInstance
	ecs: RegisterTaskDefinition
	ecs: RunTask
	ecs: StartTask
	ecs: StopTask
	ecs: SubmitAttachmentStateChanges

Awalan layanan	Tindakan
	ecs: SubmitContainerStateChange
	ecs: SubmitTaskStateChange
	ecs: UpdateCapacityProvider
	ecs: UpdateCluster
	ecs: UpdateClusterSettings
	ecs: UpdateContainerAgent
	ecs: UpdateContainerInstancesState
	ecs: UpdateService
	ecs: UpdateServicePrimaryTaskSet
	ecs: UpdateTaskProtection
	ecs: UpdateTaskSet

Awalan layanan	Tindakan
eks	eks: AssociateAccessPolicy eks: AssociateEncryptionConfig eks: AssociateIdentityProviderConfig eks: CreateAccessEntry eks: CreateAddon eks: CreateCluster eks: CreateEksAnywhereSubscription eks: CreateFargateProfile eks: CreateNodegroup eks: DeleteAccessEntry eks: DeleteAddon eks: DeleteCluster eks: DeleteEksAnywhereSubscription eks: DeleteFargateProfile eks: DeleteNodegroup eks: DeletePodIdentityAssociation eks: DeregisterCluster eks: DescribeAccessEntry eks: DescribeAddon eks: DescribeAddonConfiguration eks: DescribeAddonVersions

Awalan layanan	Tindakan
	eks: DescribeCluster eks: DescribeEksAnywhereSubscription eks: DescribeFargateProfile eks: DescribeIdentityProviderConfig eks: DescribeInsight eks: DescribeNodegroup eks: DescribePodIdentityAssociation eks: DescribeUpdate eks: DisassociateAccessPolicy eks: DisassociateIdentityProviderConfig eks: ListAccessEntries eks: ListAccessPolicies eks: ListAddons eks: ListAssociatedAccessPolicies eks: ListClusters eks: ListEksAnywhereSubscriptions eks: ListFargateProfiles eks: ListIdentityProviderConfigs eks: ListInsights eks: ListNodegroups eks: ListPodIdentityAssociations

Awalan layanan	Tindakan
	eks: ListUpdates eks: RegisterCluster eks: UpdateAccessEntry eks: UpdateAddon eks: UpdateClusterConfig eks: UpdateClusterVersion eks: UpdateEksAnywhereSubscription eks: UpdateNodegroupConfig eks: UpdateNodegroupVersion eks: UpdatePodIdentityAssociation
inferensi elastis	inferensi elastis: DescribeAcceleratorOfferings inferensi elastis: DescribeAccelerators inferensi elastis: DescribeAcceleratorTypes

Awalan layanan	Tindakan
elasticache	elastisakit: AuthorizeCacheSecurityGroupIngress elastisakit: BatchApplyUpdateAction elastisakit: BatchStopUpdateAction elastisakit: CompleteMigration elastisakit: CopyServerlessCacheSnapshot elastisakit: CopySnapshot elastisakit: CreateCacheCluster elastisakit: CreateCacheParameterGroup elastisakit: CreateCacheSecurityGroup elastisakit: CreateCacheSubnetGroup elastisakit: CreateGlobalReplicationGroup elastisakit: CreateReplicationGroup elastisakit: CreateServerlessCache elastisakit: CreateServerlessCacheSnapshot elastisakit: CreateSnapshot elastisakit: CreateUser elastisakit: CreateUserGroup elastisakit: DecreaseNodeGroupsInGlobalReplicationGroup elastisakit: DecreaseReplicaCount elastisakit: DeleteCacheCluster elastisakit: DeleteCacheParameterGroup

Awalan layanan	Tindakan
	elastisakit: DeleteCacheSecurityGroup elastisakit: DeleteCacheSubnetGroup elastisakit: DeleteGlobalReplicationGroup elastisakit: DeleteReplicationGroup elastisakit: DeleteServerlessCache elastisakit: DeleteServerlessCacheSnapshot elastisakit: DeleteSnapshot elastisakit: DeleteUser elastisakit: DeleteUserGroup elastisakit: DescribeCacheClusters elastisakit: DescribeCacheEngineVersions elastisakit: DescribeCacheParameterGroups elastisakit: DescribeCacheParameters elastisakit: DescribeCacheSecurityGroups elastisakit: DescribeCacheSubnetGroups elastisakit: DescribeEngineDefaultParameters elastisakit: DescribeEvents elastisakit: DescribeGlobalReplicationGroups elastisakit: DescribeReplicationGroups elastisakit: DescribeReservedCacheNodes elastisakit: DescribeReservedCacheNodesOfferings

Awalan layanan	Tindakan
	elastisakit: DescribeServerlessCaches elastisakit: DescribeServerlessCacheSnapshots elastisakit: DescribeServiceUpdates elastisakit: DescribeSnapshots elastisakit: DescribeUpdateActions elastisakit: DescribeUserGroups elastisakit: DescribeUsers elastisakit: DisassociateGlobalReplicationGroup elastisakit: ExportServerlessCacheSnapshot elastisakit: FailoverGlobalReplicationGroup elastisakit: IncreaseNodeGroupsInGlobalReplicationGroup elastisakit: IncreaseReplicaCount elastisakit: ListAllowedNodeTypeModifications elastisakit: ModifyCacheCluster elastisakit: ModifyCacheParameterGroup elastisakit: ModifyCacheSubnetGroup elastisakit: ModifyGlobalReplicationGroup elastisakit: ModifyReplicationGroup elastisakit: ModifyReplicationGroupShardConfiguration elastisakit: ModifyServerlessCache elastisakit: ModifyUser

Awalan layanan	Tindakan
	elastisakit: ModifyUserGroup elastisakit: PurchaseReservedCacheNodesOffering elastisakit: RebalanceSlotsInGlobalReplicationGroup elastisakit: RebootCacheCluster elastisakit: ResetCacheParameterGroup elastisakit: RevokeCacheSecurityGroupIngress elastisakit: StartMigration elastisakit: TestFailover elastisakit: TestMigration

Awalan layanan	Tindakan
tangkai beanstalk	tangkai elastis: AbortEnvironmentUpdate
	tangkai elastis: ApplyEnvironmentManagedAction
	tangkai elastis: AssociateEnvironmentOperationsRole
	Batang kacang elastis: C heckDNSAvailability
	tangkai elastis: ComposeEnvironments
	tangkai elastis: CreateApplication
	tangkai elastis: CreateApplicationVersion
	tangkai elastis: CreateConfigurationTemplate
	tangkai elastis: CreateEnvironment
	tangkai elastis: CreatePlatformVersion
	tangkai elastis: CreateStorageLocation
	tangkai elastis: DeleteApplication
	tangkai elastis: DeleteApplicationVersion
	tangkai elastis: DeleteConfigurationTemplate
	tangkai elastis: DeleteEnvironmentConfiguration
	tangkai elastis: DeletePlatformVersion
	tangkai elastis: DescribeAccountAttributes
	tangkai elastis: DescribeApplications
	tangkai elastis: DescribeApplicationVersions
	tangkai elastis: DescribeConfigurationOptions
	tangkai elastis: DescribeConfigurationSettings

Awalan layanan	Tindakan
	tangkai elastis: DescribeEnvironmentHealth
	tangkai elastis: DescribeEnvironmentManagedActionHistory
	tangkai elastis: DescribeEnvironmentManagedActions
	tangkai elastis: DescribeEnvironmentResources
	tangkai elastis: DescribeEnvironments
	tangkai elastis: DescribeEvents
	tangkai elastis: DescribeInstancesHealth
	tangkai elastis: DescribePlatformVersion
	tangkai elastis: DisassociateEnvironmentOperationsRole
	tangkai elastis: ListAvailableSolutionStacks
	tangkai elastis: ListPlatformBranches
	tangkai elastis: ListPlatformVersions
	tangkai elastis: RebuildEnvironment
	tangkai elastis: RequestEnvironmentInfo
	tangkai elastis: RestartAppServer
	tangkai elastis: RetrieveEnvironmentInfo
	tangkai elastis: SwapEnvironment CNAMEs
	tangkai elastis: TerminateEnvironment
	tangkai elastis: UpdateApplication
	tangkai elastis: UpdateApplicationResourceLifecycle
	tangkai elastis: UpdateApplicationVersion

Awalan layanan	Tindakan
	tangkai elastis: UpdateConfigurationTemplate tangkai elastis: UpdateEnvironment tangkai elastis: ValidateConfigurationSettings

Awalan layanan	Tindakan
sistem file elastis	sistem file elastis: CreateAccessPoint sistem file elastis: CreateFileSystem sistem file elastis: CreateMountTarget sistem file elastis: CreateReplicationConfiguration sistem file elastis: DeleteAccessPoint sistem file elastis: DeleteFileSystem sistem file elastis: DeleteFileSystemPolicy sistem file elastis: DeleteMountTarget sistem file elastis: DeleteReplicationConfiguration sistem file elastis: DescribeAccessPoints sistem file elastis: DescribeAccountPreferences sistem file elastis: DescribeBackupPolicy sistem file elastis: DescribeFileSystemPolicy sistem file elastis: DescribeFileSystems sistem file elastis: DescribeLifecycleConfiguration sistem file elastis: DescribeMountTargets sistem file elastis: DescribeMountTargetSecurityGroups sistem file elastis: DescribeReplicationConfigurations sistem file elastis: ModifyMountTargetSecurityGroups sistem file elastis: PutAccountPreferences sistem file elastis: PutBackupPolicy

Awalan layanan	Tindakan
	sistem file elastis: PutFileSystemPolicy sistem file elastis: PutLifecycleConfiguration sistem file elastis: UpdateFileSystem sistem file elastis: UpdateFileSystemProtection

Awalan layanan	Tindakan
elasticloadbalancing	elasticloadbalancing: AddListenerCertificates
	elasticloadbalancing: AddTrustStoreRevocations
	elasticloadbalancing: ApplySecurityGroupsToLoadBalancer
	elasticloadbalancing: AttachLoadBalancerToSubnets
	elasticloadbalancing: ConfigureHealthCheck
	elasticloadbalancing: CreateAppCookieStickinessPolicy
	ElasticLoadBalancing:C reateLBCookie StickinessPolicy
	elasticloadbalancing: CreateListener
	elasticloadbalancing: CreateLoadBalancer
	elasticloadbalancing: CreateLoadBalancerListeners
	elasticloadbalancing: CreateLoadBalancerPolicy
	elasticloadbalancing: CreateRule
	elasticloadbalancing: CreateTargetGroup
	elasticloadbalancing: CreateTrustStore
	elasticloadbalancing: DeleteListener
	elasticloadbalancing: DeleteLoadBalancer
	elasticloadbalancing: DeleteLoadBalancerListeners
	elasticloadbalancing: DeleteLoadBalancerPolicy
	elasticloadbalancing: DeleteRule
	elasticloadbalancing: DeleteSharedTrustStoreAssociation
	elasticloadbalancing: DeleteTargetGroup

Awalan layanan	Tindakan
	elasticloadbalancing: DeleteTrustStore
	elasticloadbalancing: DeregisterInstancesFromLoadBalancer
	elasticloadbalancing: DeregisterTargets
	elasticloadbalancing: DescribeAccountLimits
	elasticloadbalancing: DescribeInstanceHealth
	elasticloadbalancing: DescribeListenerAttributes
	elasticloadbalancing: DescribeListenerCertificates
	elasticloadbalancing: DescribeListeners
	elasticloadbalancing: DescribeLoadBalancerAttributes
	elasticloadbalancing: DescribeLoadBalancerPolicies
	elasticloadbalancing: DescribeLoadBalancerPolicyTypes
	elasticloadbalancing: DescribeLoadBalancers
	elasticloadbalancing: DescribeRules
	ElasticLoadBalancing:D escribeSSLPolicies
	elasticloadbalancing: DescribeTargetGroupAttributes
	elasticloadbalancing: DescribeTargetGroups
	elasticloadbalancing: DescribeTargetHealth
	elasticloadbalancing: DescribeTrustStoreAssociations
	elasticloadbalancing: DescribeTrustStoreRevocations
	elasticloadbalancing: DescribeTrustStores
	elasticloadbalancing: DetachLoadBalancerFromSubnets

Awalan layanan	Tindakan
	elasticloadbalancing: DisableAvailabilityZonesForLoadBalancer
	elasticloadbalancing: EnableAvailabilityZonesForLoadBalancer
	elasticloadbalancing: GetResourcePolicy
	elasticloadbalancing: GetTrustStoreCaCertificatesBundle
	elasticloadbalancing: GetTrustStoreRevocationContent
	elasticloadbalancing: ModifyListener
	elasticloadbalancing: ModifyLoadBalancerAttributes
	elasticloadbalancing: ModifyRule
	elasticloadbalancing: ModifyTargetGroup
	elasticloadbalancing: ModifyTargetGroupAttributes
	elasticloadbalancing: ModifyTrustStore
	elasticloadbalancing: RegisterInstancesWithLoadBalancer
	elasticloadbalancing: RegisterTargets
	elasticloadbalancing: RemoveListenerCertificates
	elasticloadbalancing: RemoveTrustStoreRevocations
	elasticloadbalancing: SetIpAddressType
	elasticloadbalancing: SetLoadBalancerListener SSLCertificate
	elasticloadbalancing: SetLoadBalancerPoliciesForBackendServer
	elasticloadbalancing: SetLoadBalancerPoliciesOfListener
	elasticloadbalancing: SetRulePriorities
	elasticloadbalancing: SetSecurityGroups

Awalan layanan	Tindakan
	elasticloadbalancing: SetSubnets
elastictranscoder	elastictranscoder: CancelJob elastictranscoder: CreateJob elastictranscoder: CreatePipeline elastictranscoder: CreatePreset elastictranscoder: DeletePipeline elastictranscoder: DeletePreset elastictranscoder: ListJobsByPipeline elastictranscoder: ListJobsByStatus elastictranscoder: ListPipelines elastictranscoder: ListPresets elastictranscoder: ReadJob elastictranscoder: ReadPipeline elastictranscoder: ReadPreset elastictranscoder: TestRole elastictranscoder: UpdatePipeline elastictranscoder: UpdatePipelineNotifications elastictranscoder: UpdatePipelineStatus

Awalan layanan	Tindakan
kontainer emr	emr-kontainer: CancelJobRun
	emr-kontainer: CreateJobTemplate
	emr-kontainer: CreateManagedEndpoint
	emr-kontainer: CreateSecurityConfiguration
	emr-kontainer: CreateVirtualCluster
	emr-kontainer: DeleteJobTemplate
	emr-kontainer: DeleteManagedEndpoint
	emr-kontainer: DeleteVirtualCluster
	emr-kontainer: DescribeJobRun
	emr-kontainer: DescribeJobTemplate
	emr-kontainer: DescribeManagedEndpoint
	emr-kontainer: DescribeSecurityConfiguration
	emr-kontainer: DescribeVirtualCluster
	emr-kontainer: GetManagedEndpointSessionCredentials
	emr-kontainer: ListJobRuns
	emr-kontainer: ListJobTemplates
	emr-kontainer: ListManagedEndpoints
	emr-kontainer: ListSecurityConfigurations
	emr-kontainer: ListVirtualClusters
	emr-kontainer: StartJobRun

Awalan layanan	Tindakan
emr-tanpa server	emr-tanpa server: CancelJobRun emr-tanpa server: CreateApplication emr-tanpa server: DeleteApplication emr-tanpa server: GetApplication emr-tanpa server: GetDashboardForJobRun emr-tanpa server: GetJobRun emr-tanpa server: ListApplications emr-tanpa server: ListJobRunAttempts emr-tanpa server: ListJobRuns emr-tanpa server: StartApplication emr-tanpa server: StartJobRun emr-tanpa server: StopApplication emr-tanpa server: UpdateApplication

Awalan layanan	Tindakan
es	es: AcceptInboundConnection
	es: AcceptInboundCrossClusterSearchConnection
	es: AssociatePackage
	es: AuthorizeVpcEndpointAccess
	es: CancelElasticsearchServiceSoftwareUpdate
	es: CancelServiceSoftwareUpdate
	es: CreateDomain
	es: CreateElasticsearchDomain
	es: CreateOutboundConnection
	es: CreateOutboundCrossClusterSearchConnection
	es: CreatePackage
	es: CreateVpcEndpoint
	es: DeleteDomain
	es: DeleteElasticsearchDomain
	es: DeleteElasticsearchServiceRole
	es: DeleteInboundConnection
	es: DeleteInboundCrossClusterSearchConnection
	es: DeleteOutboundConnection
	es: DeleteOutboundCrossClusterSearchConnection
	es: DeletePackage
	es: DeleteVpcEndpoint

Awalan layanan	Tindakan
	es: DescribeDomain
	es: DescribeDomainAutoTunes
	es: DescribeDomainChangeProgress
	es: DescribeDomainConfig
	es: DescribeDomainHealth
	es: DescribeDomainNodes
	es: DescribeDomains
	es: DescribeDryRunProgress
	es: DescribeElasticsearchDomain
	es: DescribeElasticsearchDomainConfig
	es: DescribeElasticsearchDomains
	es: DescribeElasticsearchInstanceTypeLimits
	es: DescribeInboundConnections
	es: DescribeInboundCrossClusterSearchConnections
	es: DescribeInstanceTypeLimits
	es: DescribeOutboundConnections
	es: DescribeOutboundCrossClusterSearchConnections
	es: DescribePackages
	es: DescribeReservedElasticsearchInstanceOfferings
	es: DescribeReservedElasticsearchInstances
	es: DescribeReservedInstanceOfferings

Awalan layanan	Tindakan
	es: DescribeReservedInstances
	es: DescribeVpcEndpoints
	es: DissociatePackage
	es: GetCompatibleElasticsearchVersions
	es: GetCompatibleVersions
	es: GetDataSource
	es: GetDomainMaintenanceStatus
	es: GetPackageVersionHistory
	es: GetUpgradeHistory
	es: GetUpgradeStatus
	es: ListDataSources
	es: ListDomainNames
	es: ListDomainsForPackage
	es: ListElasticsearchInstanceTypes
	es: ListElasticsearchVersions
	es: ListInstanceTypeDetails
	es: ListPackagesForDomain
	es: ListScheduledActions
	es: ListVersions
	es: ListVpcEndpointAccess
	es: ListVpcEndpoints

Awalan layanan	Tindakan
	es: ListVpcEndpointsForDomain
	es: PurchaseReservedElasticsearchInstanceOffering
	es: PurchaseReservedInstanceOffering
	es: RejectInboundConnection
	es: RejectInboundCrossClusterSearchConnection
	es: RevokeVpcEndpointAccess
	es: StartDomainMaintenance
	es: StartElasticsearchServiceSoftwareUpdate
	es: StartServiceSoftwareUpdate
	es: UpdateDataSource
	es: UpdateDomainConfig
	es: UpdateElasticsearchDomainConfig
	es: UpdatePackage
	es: UpdateScheduledAction
	es: UpdateVpcEndpoint
	es: UpgradeDomain
	es: UpgradeElasticsearchDomain

Awalan layanan	Tindakan
peristiwa	peristiwa: ActivateEventSource peristiwa: CancelReplay peristiwa: CreateApiDestination peristiwa: CreateArchive peristiwa: CreateConnection peristiwa: CreateEndpoint peristiwa: CreateEventBus peristiwa: CreatePartnerEventSource peristiwa: DeactivateEventSource peristiwa: DeauthorizeConnection peristiwa: DeleteApiDestination peristiwa: DeleteArchive peristiwa: DeleteConnection peristiwa: DeleteEndpoint peristiwa: DeleteEventBus peristiwa: DeletePartnerEventSource peristiwa: DeleteRule peristiwa: DescribeApiDestination peristiwa: DescribeArchive peristiwa: DescribeConnection peristiwa: DescribeEndpoint

Awalan layanan	Tindakan
	peristiwa: DescribeEventBus peristiwa: DescribeEventSource peristiwa: DescribePartnerEventSource peristiwa: DescribeReplay peristiwa: DescribeRule peristiwa: DisableRule peristiwa: EnableRule peristiwa: ListApiDestinations peristiwa: ListArchives peristiwa: ListConnections peristiwa: ListEndpoints peristiwa: ListEventBuses peristiwa: ListEventSources peristiwa: ListPartnerEventSourceAccounts peristiwa: ListPartnerEventSources peristiwa: ListReplays peristiwa: ListRuleNamesByTarget peristiwa: ListRules peristiwa: ListTargetsByRule peristiwa: PutPermission peristiwa: PutRule

Awalan layanan	Tindakan
	peristiwa: PutTargets peristiwa: RemovePermission peristiwa: RemoveTargets peristiwa: StartReplay peristiwa: TestEventPattern peristiwa: UpdateApiDestination peristiwa: UpdateArchive peristiwa: UpdateConnection peristiwa: UpdateEndpoint peristiwa: UpdateEventBus

Awalan layanan	Tindakan
ternyata	jelas: CreateExperiment
	jelas: CreateFeature
	jelas: CreateLaunch
	jelas: CreateProject
	jelas: CreateSegment
	jelas: DeleteExperiment
	jelas: DeleteFeature
	jelas: DeleteLaunch
	jelas: DeleteProject
	jelas: DeleteSegment
	jelas: GetExperiment
	jelas: GetExperimentResults
	jelas: GetFeature
	jelas: GetLaunch
	jelas: GetProject
	jelas: GetSegment
	jelas: ListExperiments
	jelas: ListFeatures
	jelas: ListLaunches
	jelas: ListProjects
	jelas: ListSegmentReferences

Awalan layanan	Tindakan
	jelas: ListSegments
	jelas: StartExperiment
	jelas: StartLaunch
	jelas: StopExperiment
	jelas: StopLaunch
	jelas: TestSegmentPattern
	jelas: UpdateExperiment
	jelas: UpdateFeature
	jelas: UpdateLaunch
	jelas: UpdateProject
	jelas: UpdateProjectDataDelivery

Awalan layanan	Tindakan
ruang finspace	ruang finspace: CreateEnvironment ruang finspace: CreateKxChangeset ruang finspace: CreateKxCluster ruang finspace: CreateKxDatabase ruang finspace: CreateKxDataview ruang finspace: CreateKxEnvironment ruang finspace: CreateKxScalingGroup ruang finspace: CreateKxUser ruang finspace: CreateKxVolume ruang finspace: CreateUser ruang finspace: DeleteEnvironment ruang finspace: DeleteKxCluster ruang finspace: DeleteKxClusterNode ruang finspace: DeleteKxDatabase ruang finspace: DeleteKxDataview ruang finspace: DeleteKxEnvironment ruang finspace: DeleteKxScalingGroup ruang finspace: DeleteKxUser ruang finspace: DeleteKxVolume ruang finspace: GetEnvironment ruang finspace: GetKxChangeset

Awalan layanan	Tindakan
	ruang finspace: GetKxCluster ruang finspace: GetKxConnectionString ruang finspace: GetKxDatabase ruang finspace: GetKxDataview ruang finspace: GetKxEnvironment ruang finspace: GetKxScalingGroup ruang finspace: GetKxUser ruang finspace: GetKxVolume ruang finspace: GetLoadSampleDataSetGroupIntoEnvironmentStatus ruang finspace: GetUser ruang finspace: ListEnvironments ruang finspace: ListKxChangesets ruang finspace: ListKxClusterNodes ruang finspace: ListKxClusters ruang finspace: ListKxDatabases ruang finspace: ListKxDataviews ruang finspace: ListKxEnvironments ruang finspace: ListKxScalingGroups ruang finspace: ListKxUsers ruang finspace: ListKxVolumes

Awalan layanan	Tindakan
	ruang finspace: ListUsers ruang finspace: LoadSampleDataSetGroupIntoEnvironment ruang finspace: ResetUserPassword ruang finspace: UpdateEnvironment ruang finspace: UpdateKxClusterCodeConfiguration ruang finspace: UpdateKxClusterDatabases ruang finspace: UpdateKxDatabase ruang finspace: UpdateKxDataview ruang finspace: UpdateKxEnvironment ruang finspace: UpdateKxEnvironmentNetwork ruang finspace: UpdateKxUser ruang finspace: UpdateKxVolume ruang finspace: UpdateUser
firehose	selang api: CreateDeliveryStream selang api: DeleteDeliveryStream selang api: DescribeDeliveryStream selang api: ListDeliveryStreams selang api: StartDeliveryStreamEncryption selang api: StopDeliveryStreamEncryption selang api: UpdateDestination

Awalan layanan	Tindakan
fis	fis: CreateExperimentTemplate fis: CreateTargetAccountConfiguration fis: DeleteExperimentTemplate fis: DeleteTargetAccountConfiguration fis: GetAction fis: GetExperiment fis: GetExperimentTargetAccountConfiguration fis: GetExperimentTemplate fis: GetSafetyLever fis: GetTargetAccountConfiguration fis: GetTargetResourceType fis: ListActions fis: ListExperimentResolvedTargets fis: ListExperiments fis: ListExperimentTargetAccountConfigurations fis: ListExperimentTemplates fis: ListTargetAccountConfigurations fis: ListTargetResourceTypes fis: StartExperiment fis: StopExperiment fis: UpdateExperimentTemplate

Awalan layanan	Tindakan
	fis: UpdateSafetyLeverState fis: UpdateTargetAccountConfiguration

Awalan layanan	Tindakan
fms	fms: AssociateAdminAccount
	fms: AssociateThirdPartyFirewall
	fms: BatchAssociateResource
	fms: BatchDisassociateResource
	fms: DeleteAppsList
	fms: DeleteNotificationChannel
	fms: DeletePolicy
	fms: DeleteProtocolsList
	fms: DeleteResourceSet
	fms: DisassociateAdminAccount
	fms: DisassociateThirdPartyFirewall
	fms: GetAdminAccount
	fms: GetAdminScope
	fms: GetAppsList
	fms: GetComplianceDetail
	fms: GetNotificationChannel
	fms: GetPolicy
	fms: GetProtectionStatus
	fms: GetProtocolsList
	fms: GetResourceSet
	fms: GetThirdPartyFirewallAssociationStatus

Awalan layanan	Tindakan
	fms: GetViolationDetails
	fms: ListAdminAccountsForOrganization
	fms: ListAdminsManagingAccount
	fms: ListAppsLists
	fms: ListComplianceStatus
	fms: ListDiscoveredResources
	fms: ListMemberAccounts
	fms: ListPolicies
	fms: ListProtocolsLists
	fms: ListResourceSetResources
	fms: ListResourceSets
	fms: ListThirdPartyFirewallFirewallPolicies
	fms: PutAdminAccount
	fms: PutAppsList
	fms: PutNotificationChannel
	fms: PutPolicy
	fms: PutProtocolsList
	fms: PutResourceSet

Awalan layanan	Tindakan
detektor penipuan	detektor penipuan: BatchCreateVariable detektor penipuan: BatchGetVariable detektor penipuan: CancelBatchImportJob detektor penipuan: CancelBatchPredictionJob detektor penipuan: CreateBatchImportJob detektor penipuan: CreateBatchPredictionJob detektor penipuan: CreateDetectorVersion detektor penipuan: CreateList detektor penipuan: CreateModel detektor penipuan: CreateModelVersion detektor penipuan: CreateRule detektor penipuan: CreateVariable detektor penipuan: DeleteBatchImportJob detektor penipuan: DeleteBatchPredictionJob detektor penipuan: DeleteDetector detektor penipuan: DeleteDetectorVersion detektor penipuan: DeleteEntityType detektor penipuan: DeleteEvent detektor penipuan: DeleteEventsByEventType detektor penipuan: DeleteEventType detektor penipuan: DeleteExternalModel

Awalan layanan	Tindakan
	detektor penipuan: DeleteLabel detektor penipuan: DeleteList detektor penipuan: DeleteModel detektor penipuan: DeleteModelVersion detektor penipuan: DeleteOutcome detektor penipuan: DeleteRule detektor penipuan: DeleteVariable detektor penipuan: DescribeDetector detektor penipuan: DescribeModelVersions detektor penipuan: GetBatchImportJobs detektor penipuan: GetBatchPredictionJobs detektor penipuan: GetDeleteEventsByEventTypeStatus detektor penipuan: GetDetectors detektor penipuan: GetDetectorVersion detektor penipuan: GetEntityTypes detektor penipuan: GetEvent detektor penipuan: GetEventPrediction detektor penipuan: GetEventPredictionMetadata detektor penipuan: GetEventTypes detektor penipuan: GetExternalModels Detektor Penipu:G Kunci etKMSEncryption

Awalan layanan	Tindakan
	detektor penipuan: GetLabels detektor penipuan: GetListElements detektor penipuan: GetListsMetadata detektor penipuan: GetModels detektor penipuan: GetModelVersion detektor penipuan: GetOutcomes detektor penipuan: GetRules detektor penipuan: GetVariables detektor penipuan: ListEventPredictions detektor penipuan: PutDetector detektor penipuan: PutEntityType detektor penipuan: PutEventType detektor penipuan: PutExternalModel Detektor Penipu:Kunci P utKMSEncryption detektor penipuan: PutLabel detektor penipuan: PutOutcome detektor penipuan: SendEvent detektor penipuan: UpdateDetectorVersion detektor penipuan: UpdateDetectorVersionMetadata detektor penipuan: UpdateDetectorVersionStatus detektor penipuan: UpdateEventLabel

Awalan layanan	Tindakan
	detektor penipuan: UpdateList detektor penipuan: UpdateModel detektor penipuan: UpdateModelVersion detektor penipuan: UpdateModelVersionStatus detektor penipuan: UpdateRuleMetadata detektor penipuan: UpdateRuleVersion detektor penipuan: UpdateVariable

Awalan layanan	Tindakan
fsx	fsx: AssociateFileSystemAliases
	fsx: CancelDataRepositoryTask
	fsx: CopyBackup
	fsx: CreateDataRepositoryTask
	fsx: CreateFileCache
	fsx: CreateFileSystem
	fsx: CreateFileSystemFromBackup
	fsx: CreateSnapshot
	fsx: CreateStorageVirtualMachine
	fsx: CreateVolume
	fsx: CreateVolumeFromBackup
	fsx: DeleteBackup
	fsx: DeleteFileCache
	fsx: DeleteFileSystem
	fsx: DeleteSnapshot
	fsx: DeleteStorageVirtualMachine
	fsx: DeleteVolume
	fsx: DescribeBackups
	fsx: DescribeDataRepositoryAssociations
	fsx: DescribeDataRepositoryTasks
	fsx: DescribeFileCaches

Awalan layanan	Tindakan
	fsx: DescribeFileSystemAliases
	fsx: DescribeFileSystems
	fsx: DescribeSharedVpcConfiguration
	fsx: DescribeSnapshots
	fsx: DescribeStorageVirtualMachines
	fsx: DescribeVolumes
	fsx: DisassociateFileSystemAliases
	fsx: ReleaseFileSystemNfs V3Locks
	fsx: RestoreVolumeFromSnapshot
	fsx: StartMisconfiguredStateRecovery
	fsx: UpdateDataRepositoryAssociation
	fsx: UpdateFileCache
	fsx: UpdateFileSystem
	fsx: UpdateSharedVpcConfiguration
	fsx: UpdateSnapshot
	fsx: UpdateStorageVirtualMachine
	fsx: UpdateVolume

Awalan layanan	Tindakan
gamelift	gamelift: AcceptMatch gamelift: ClaimGameServer gamelift: CreateAlias gamelift: CreateBuild gamelift: CreateContainerGroupDefinition gamelift: CreateFleet gamelift: CreateFleetLocations gamelift: CreateGameServerGroup gamelift: CreateGameSession gamelift: CreateGameSessionQueue gamelift: CreateLocation gamelift: CreateMatchmakingConfiguration gamelift: CreateMatchmakingRuleSet gamelift: CreatePlayerSession gamelift: CreatePlayerSessions gamelift: CreateScript gamelift: CreateVpcPeeringAuthorization gamelift: CreateVpcPeeringConnection gamelift: DeleteAlias gamelift: DeleteBuild gamelift: DeleteContainerGroupDefinition

Awalan layanan	Tindakan
	gamelift: DeleteFleet gamelift: DeleteFleetLocations gamelift: DeleteGameServerGroup gamelift: DeleteGameSessionQueue gamelift: DeleteLocation gamelift: DeleteMatchmakingConfiguration gamelift: DeleteMatchmakingRuleSet gamelift: DeleteScalingPolicy gamelift: DeleteScript gamelift: DeleteVpcPeeringAuthorization gamelift: DeleteVpcPeeringConnection gamelift: DeregisterCompute gamelift: DeregisterGameServer gamelift: DescribeAlias gamelift: DescribeBuild gamelift: DescribeCompute gamelift: DescribeContainerGroupDefinition GameLift: jelaskan EC2InstanceLimits gamelift: DescribeFleetAttributes gamelift: DescribeFleetCapacity gamelift: DescribeFleetEvents

Awalan layanan	Tindakan
	gamelift: DescribeFleetLocationAttributes
	gamelift: DescribeFleetLocationCapacity
	gamelift: DescribeFleetLocationUtilization
	gamelift: DescribeFleetPortSettings
	gamelift: DescribeFleetUtilization
	gamelift: DescribeGameServer
	gamelift: DescribeGameServerGroup
	gamelift: DescribeGameServerInstances
	gamelift: DescribeGameSessionDetails
	gamelift: DescribeGameSessionPlacement
	gamelift: DescribeGameSessionQueues
	gamelift: DescribeGameSessions
	gamelift: DescribeInstances
	gamelift: DescribeMatchmaking
	gamelift: DescribeMatchmakingConfigurations
	gamelift: DescribeMatchmakingRuleSets
	gamelift: DescribePlayerSessions
	gamelift: DescribeRuntimeConfiguration
	gamelift: DescribeScalingPolicies
	gamelift: DescribeScript
	gamelift: DescribeVpcPeeringAuthorizations

Awalan layanan	Tindakan
	gamelift: DescribeVpcPeeringConnections gamelift: GetComputeAccess gamelift: GetComputeAuthToken gamelift: GetGameSessionLogUrl gamelift: GetInstanceAccess gamelift: ListAliases gamelift: ListBuilds gamelift: ListCompute gamelift: ListContainerGroupDefinitions gamelift: ListFleets gamelift: ListGameServerGroups gamelift: ListGameServers gamelift: ListLocations gamelift: ListScripts gamelift: PutScalingPolicy gamelift: RegisterCompute gamelift: RegisterGameServer gamelift: RequestUploadCredentials gamelift: ResolveAlias gamelift: ResumeGameServerGroup gamelift: SearchGameSessions

Awalan layanan	Tindakan
	gamelift: StartFleetActions gamelift: StartGameSessionPlacement gamelift: StartMatchBackfill gamelift: StartMatchmaking gamelift: StopFleetActions gamelift: StopGameSessionPlacement gamelift: StopMatchmaking gamelift: SuspendGameServerGroup gamelift: UpdateAlias gamelift: UpdateBuild gamelift: UpdateFleetAttributes gamelift: UpdateFleetCapacity gamelift: UpdateFleetPortSettings gamelift: UpdateGameServer gamelift: UpdateGameServerGroup gamelift: UpdateGameSession gamelift: UpdateGameSessionQueue gamelift: UpdateMatchmakingConfiguration gamelift: UpdateRuntimeConfiguration gamelift: UpdateScript gamelift: ValidateMatchmakingRuleSet

Awalan layanan	Tindakan
geo	geografis: AssociateTrackerConsumer
	geografis: BatchDeleteDevicePositionHistory
	geografis: BatchDeleteGeofence
	geografis: BatchEvaluateGeofences
	geografis: BatchGetDevicePosition
	geografis: BatchPutGeofence
	geografis: BatchUpdateDevicePosition
	geografis: CalculateRoute
	geografis: CalculateRouteMatrix
	geografis: CreateGeofenceCollection
	geografis: CreateMap
	geografis: CreatePlaceIndex
	geografis: CreateRouteCalculator
	geografis: CreateTracker
	geografis: DeleteGeofenceCollection
	geografis: DeleteKey
	geografis: DeleteMap
	geografis: DeletePlaceIndex
	geografis: DeleteRouteCalculator
	geografis: DeleteTracker
	geografis: DescribeGeofenceCollection

Awalan layanan	Tindakan
	geografis: DescribeKey
	geografis: DescribeMap
	geografis: DescribePlaceIndex
	geografis: DescribeRouteCalculator
	geografis: DescribeTracker
	geografis: DisassociateTrackerConsumer
	geografis: ForecastGeofenceEvents
	geografis: GetDevicePosition
	geografis: GetDevicePositionHistory
	geografis: GetGeofence
	geografis: GetMapGlyphs
	geografis: GetMapSprites
	geografis: GetMapStyleDescriptor
	geografis: GetMapTile
	geografis: GetPlace
	geografis: ListDevicePositions
	geografis: ListGeofenceCollections
	geografis: ListGeofences
	geografis: ListKeys
	geografis: ListMaps
	geografis: ListPlaceIndexes

Awalan layanan	Tindakan
	geografis: ListRouteCalculators
	geografis: ListTrackerConsumers
	geografis: ListTrackers
	geografis: PutGeofence
	geografis: SearchPlaceIndexForPosition
	geografis: SearchPlaceIndexForSuggestions
	geografis: SearchPlaceIndexForText
	geografis: UpdateGeofenceCollection
	geografis: UpdateKey
	geografis: UpdateMap
	geografis: UpdatePlaceIndex
	geografis: UpdateRouteCalculator
	geografis: UpdateTracker
	geografis: VerifyDevicePosition

Awalan layanan	Tindakan
gletser	gletser: AbortMultipartUpload gletser: AbortVaultLock gletser: CompleteMultipartUpload gletser: CompleteVaultLock gletser: CreateVault gletser: DeleteArchive gletser: DeleteVault gletser: DeleteVaultAccessPolicy gletser: DeleteVaultNotifications gletser: DescribeJob gletser: DescribeVault gletser: GetDataRetrievalPolicy gletser: GetJobOutput gletser: GetVaultAccessPolicy gletser: GetVaultLock gletser: GetVaultNotifications gletser: InitiateJob gletser: InitiateMultipartUpload gletser: InitiateVaultLock gletser: ListJobs gletser: ListMultipartUploads

Awalan layanan	Tindakan
	gletser: ListParts gletser: ListProvisionedCapacity gletser: ListVaults gletser: PurchaseProvisionedCapacity gletser: SetDataRetrievalPolicy gletser: SetVaultAccessPolicy gletser: SetVaultNotifications gletser: UploadArchive gletser: UploadMultipartPart

Awalan layanan	Tindakan
grafana	grafana: AssociateLicense
	grafana: CreateWorkspace
	grafana: CreateWorkspaceApiKey
	grafana: CreateWorkspaceServiceAccount
	grafana: CreateWorkspaceServiceAccountToken
	grafana: DeleteWorkspace
	grafana: DeleteWorkspaceApiKey
	grafana: DeleteWorkspaceServiceAccount
	grafana: DeleteWorkspaceServiceAccountToken
	grafana: DescribeWorkspace
	grafana: DescribeWorkspaceAuthentication
	grafana: DescribeWorkspaceConfiguration
	grafana: DisassociateLicense
	grafana: ListPermissions
	grafana: ListVersions
	grafana: ListWorkspaces
	grafana: ListWorkspaceServiceAccounts
	grafana: ListWorkspaceServiceAccountTokens
	grafana: UpdatePermissions
	grafana: UpdateWorkspace
	grafana: UpdateWorkspaceAuthentication

Awalan layanan	Tindakan
	grafana: UpdateWorkspaceConfiguration

Awalan layanan	Tindakan
greengrass	greengrass: AssociateRoleToGroup greengrass: AssociateServiceRoleToAccount greengrass: BatchAssociateClientDeviceWithCoreDevice greengrass: BatchDisassociateClientDeviceFromCoreDevice greengrass: CancelDeployment greengrass: CreateComponentVersion greengrass: CreateConnectorDefinition greengrass: CreateConnectorDefinitionVersion greengrass: CreateCoreDefinition greengrass: CreateCoreDefinitionVersion greengrass: CreateDeployment greengrass: CreateDeviceDefinition greengrass: CreateDeviceDefinitionVersion greengrass: CreateFunctionDefinition greengrass: CreateFunctionDefinitionVersion greengrass: CreateGroup greengrass: CreateGroupCertificateAuthority greengrass: CreateGroupVersion greengrass: CreateLoggerDefinition greengrass: CreateLoggerDefinitionVersion greengrass: CreateResourceDefinition

Awalan layanan	Tindakan
	greengrass: CreateResourceDefinitionVersion greengrass: CreateSoftwareUpdateJob greengrass: CreateSubscriptionDefinition greengrass: CreateSubscriptionDefinitionVersion greengrass: DeleteComponent greengrass: DeleteConnectorDefinition greengrass: DeleteCoreDefinition greengrass: DeleteCoreDevice greengrass: DeleteDeployment greengrass: DeleteDeviceDefinition greengrass: DeleteFunctionDefinition greengrass: DeleteGroup greengrass: DeleteLoggerDefinition greengrass: DeleteResourceDefinition greengrass: DeleteSubscriptionDefinition greengrass: DescribeComponent greengrass: DisassociateRoleFromGroup greengrass: DisassociateServiceRoleFromAccount greengrass: GetAssociatedRole greengrass: GetBulkDeploymentStatus greengrass: GetComponent

Awalan layanan	Tindakan
	greengrass: GetComponentVersionArtifact greengrass: GetConnectivityInfo greengrass: GetConnectorDefinition greengrass: GetConnectorDefinitionVersion greengrass: GetCoreDefinition greengrass: GetCoreDefinitionVersion greengrass: GetCoreDevice greengrass: GetDeployment greengrass: GetDeploymentStatus greengrass: GetDeviceDefinition greengrass: GetDeviceDefinitionVersion greengrass: GetFunctionDefinition greengrass: GetFunctionDefinitionVersion greengrass: GetGroup greengrass: GetGroupCertificateAuthority greengrass: GetGroupCertificateConfiguration greengrass: GetGroupVersion greengrass: GetLoggerDefinition greengrass: GetLoggerDefinitionVersion greengrass: GetResourceDefinition greengrass: GetResourceDefinitionVersion

Awalan layanan	Tindakan
	greengrass: GetServiceRoleForAccount greengrass: GetSubscriptionDefinition greengrass: GetSubscriptionDefinitionVersion greengrass: GetThingRuntimeConfiguration greengrass: ListBulkDeploymentDetailedReports greengrass: ListBulkDeployments greengrass: ListClientDevicesAssociatedWithCoreDevice greengrass: ListComponents greengrass: ListComponentVersions greengrass: ListConnectorDefinitions greengrass: ListConnectorDefinitionVersions greengrass: ListCoreDefinitions greengrass: ListCoreDefinitionVersions greengrass: ListCoreDevices greengrass: ListDeployments greengrass: ListDeviceDefinitions greengrass: ListDeviceDefinitionVersions greengrass: ListEffectiveDeployments greengrass: ListFunctionDefinitions greengrass: ListFunctionDefinitionVersions greengrass: ListGroupCertificateAuthorities

Awalan layanan	Tindakan
	greengrass: ListGroups greengrass: ListGroupVersions greengrass: ListInstalledComponents greengrass: ListLoggerDefinitions greengrass: ListLoggerDefinitionVersions greengrass: ListResourceDefinitions greengrass: ListResourceDefinitionVersions greengrass: ListSubscriptionDefinitions greengrass: ListSubscriptionDefinitionVersions greengrass: ResetDeployments greengrass: StartBulkDeployment greengrass: StopBulkDeployment greengrass: UpdateConnectivityInfo greengrass: UpdateConnectorDefinition greengrass: UpdateCoreDefinition greengrass: UpdateDeviceDefinition greengrass: UpdateFunctionDefinition greengrass: UpdateGroup greengrass: UpdateGroupCertificateConfiguration greengrass: UpdateLoggerDefinition greengrass: UpdateResourceDefinition

Awalan layanan	Tindakan
	greengrass: UpdateSubscriptionDefinition greengrass: UpdateThingRuntimeConfiguration

Awalan layanan	Tindakan
stasiun tanah	stasiun tanah: CancelContact
	stasiun tanah: CreateConfig
	stasiun tanah: CreateDataflowEndpointGroup
	stasiun tanah: CreateEphemeris
	stasiun tanah: CreateMissionProfile
	stasiun tanah: DeleteConfig
	stasiun tanah: DeleteDataflowEndpointGroup
	stasiun tanah: DeleteEphemeris
	stasiun tanah: DeleteMissionProfile
	stasiun tanah: DescribeContact
	stasiun tanah: DescribeEphemeris
	stasiun tanah: GetConfig
	stasiun tanah: GetDataflowEndpointGroup
	stasiun tanah: GetMinuteUsage
	stasiun tanah: GetMissionProfile
	stasiun tanah: GetSatellite
	stasiun tanah: ListConfigs
	stasiun tanah: ListContacts
	stasiun tanah: ListDataflowEndpointGroups
	stasiun tanah: ListEphemerides
	stasiun tanah: ListGroundStations

Awalan layanan	Tindakan
	stasiun tanah: ListMissionProfiles
	stasiun tanah: ListSatellites
	stasiun tanah: RegisterAgent
	stasiun tanah: ReserveContact
	stasiun tanah: UpdateAgentStatus
	stasiun tanah: UpdateConfig
	stasiun tanah: UpdateEphemeris
	stasiun tanah: UpdateMissionProfile

Awalan layanan	Tindakan
tugas jaga	tugas jaga: AcceptAdministratorInvitation tugas jaga: AcceptInvitation tugas jaga: ArchiveFindings tugas jaga: CreateDetector tugas jaga: CreateFilter Perwalian: C reateIPSet tugas jaga: CreateMalwareProtectionPlan tugas jaga: CreateMembers tugas jaga: CreatePublishingDestination tugas jaga: CreateSampleFindings tugas jaga: CreateThreatIntelSet tugas jaga: DeclineInvitations tugas jaga: DeleteDetector tugas jaga: DeleteFilter tugas jaga: DeleteInvitations Perwalian: D eteIPSet tugas jaga: DeleteMalwareProtectionPlan tugas jaga: DeleteMembers tugas jaga: DeletePublishingDestination tugas jaga: DeleteThreatIntelSet tugas jaga: DescribeMalwareScans

Awalan layanan	Tindakan
	tugas jaga: DescribeOrganizationConfiguration tugas jaga: DescribePublishingDestination tugas jaga: DisableOrganizationAdminAccount tugas jaga: DisassociateFromAdministratorAccount tugas jaga: DisassociateFromMasterAccount tugas jaga: DisassociateMembers tugas jaga: EnableOrganizationAdminAccount tugas jaga: GetAdministratorAccount tugas jaga: GetCoverageStatistics tugas jaga: GetDetector tugas jaga: GetFilter tugas jaga: GetFindings tugas jaga: GetFindingsStatistics tugas jaga: GetInvitationsCount GuardDuty: G etIPSet tugas jaga: GetMalwareProtectionPlan tugas jaga: GetMalwareScanSettings tugas jaga: GetMasterAccount tugas jaga: GetMemberDetectors tugas jaga: GetMembers tugas jaga: GetOrganizationStatistics

Awalan layanan	Tindakan
	tugas jaga: GetRemainingFreeTrialDays tugas jaga: GetThreatIntelSet tugas jaga: GetUsageStatistics tugas jaga: InviteMembers tugas jaga: ListCoverage tugas jaga: ListDetectors tugas jaga: ListFilters tugas jaga: ListFindings tugas jaga: ListInvitations Perwalian: L istIPSets tugas jaga: ListMalwareProtectionPlans tugas jaga: ListMembers tugas jaga: ListOrganizationAdminAccounts tugas jaga: ListPublishingDestinations tugas jaga: ListThreatIntelSets tugas jaga: SendSecurityTelemetry tugas jaga: StartMalwareScan tugas jaga: StartMonitoringMembers tugas jaga: StopMonitoringMembers tugas jaga: UnarchiveFindings tugas jaga: UpdateDetector

Awalan layanan	Tindakan
	tugas jaga: UpdateFilter tugas jaga: UpdateFindingsFeedback GuardDuty: U pdateIPSet tugas jaga: UpdateMalwareProtectionPlan tugas jaga: UpdateMalwareScanSettings tugas jaga: UpdateMemberDetectors tugas jaga: UpdateOrganizationConfiguration tugas jaga: UpdatePublishingDestination tugas jaga: UpdateThreatIntelSet

Awalan layanan	Tindakan
healthlake	Healthlake:C reateFHIRDatastore
	Healthlake: CreateResource
	Healthlake:D eleteFHIRDatastore
	Healthlake: DeleteResource
	Healthlake:D escribeFHIRDatastore
	escribeFHIRExportHealthlake:D Job
	escribeFHIRImportHealthlake:D Job
	Healthlake: GetCapabilities
	Healthlake:L istFHIRDatastores
	istFHIRExportHealthlake:L Pekerjaan
	istFHIRImportHealthlake:L Pekerjaan
	Healthlake: ReadResource
	Healthlake: SearchEverything
	Healthlake: SearchWithGet
	Healthlake: SearchWithPost
	tartFHIRExportHealthlake:s Job
	tartFHIRImportHealthlake:s Job
	Healthlake: UpdateResource

Awalan layanan	Tindakan
kode madu	kode madu: BatchCreateTableRows kode madu: BatchDeleteTableRows kode madu: BatchUpdateTableRows kode madu: BatchUpsertTableRows kode madu: DescribeTableDataImportJob kode madu: GetScreenData kode madu: InvokeScreenAutomation kode madu: ListTableColumns kode madu: ListTableRows kode madu: ListTables kode madu: QueryTableRows kode madu: StartTableDataImportJob

Awalan layanan	Tindakan
iam	saya: AddClient IDToOpenIDConnectProvider
	saya: AddRoleToInstanceProfile
	saya: AddUserToGroup
	saya: AttachGroupPolicy
	saya: AttachRolePolicy
	saya: AttachUserPolicy
	saya: ChangePassword
	saya: CreateAccessKey
	saya: CreateAccountAlias
	saya: CreateGroup
	saya: CreateInstanceProfile
	saya: CreateLoginProfile
	saya: CreateOpen IDConnectProvider
	saya: CreatePolicy
	saya: CreatePolicyVersion
	saya: CreateRole
	Iam:C reateSAMLProvider
	saya: CreateServiceLinkedRole
	saya: CreateServiceSpecificCredential
	saya: CreateUser
	saya: CreateVirtual MFADevice

Awalan layanan	Tindakan
	Iam:D eactivateMFADevice
	saya: DeleteAccessKey
	saya: DeleteAccountAlias
	saya: DeleteAccountPasswordPolicy
	saya: DeleteCloudFrontPublicKey
	saya: DeleteGroup
	saya: DeleteGroupPolicy
	saya: DeleteInstanceProfile
	saya: DeleteLoginProfile
	saya: DeleteOpen IDConnectProvider
	saya: DeletePolicy
	saya: DeletePolicyVersion
	saya: DeleteRole
	saya: DeleteRolePermissionsBoundary
	saya: DeleteRolePolicy
	Iam:D eleteSAMLProvider
	saya: DeleteServerCertificate
	saya: DeleteServiceLinkedRole
	saya: DeleteServiceSpecificCredential
	saya: DeleteSigningCertificate
	Iam:D Kunci eleteSSHPublic

Awalan layanan	Tindakan
	saya: DeleteUser
	saya: DeleteUserPermissionsBoundary
	saya: DeleteUserPolicy
	saya: DeleteVirtual MFADevice
	saya: DetachGroupPolicy
	saya: DetachRolePolicy
	saya: DetachUserPolicy
	Iam:E nableMFADevice
	saya: GenerateCredentialReport
	saya: GenerateOrganizationsAccessReport
	saya: GenerateServiceLastAccessedDetails
	saya: GetAccessKeyLastUsed
	saya: GetAccountAuthorizationDetails
	saya: GetAccountEmailAddress
	saya: GetAccountName
	saya: GetAccountPasswordPolicy
	saya: GetAccountSummary
	saya: GetCloudFrontPublicKey
	saya: GetContextKeysForCustomPolicy
	saya: GetContextKeysForPrincipalPolicy
	saya: GetCredentialReport

Awalan layanan	Tindakan
	saya: GetGroup
	saya: GetGroupPolicy
	saya: GetInstanceProfile
	saya: GetLoginProfile
	Iam:g etMFADevice
	saya: GetOpen IDConnectProvider
	saya: GetOrganizationsAccessReport
	saya: GetPolicy
	saya: GetPolicyVersion
	saya: GetRole
	saya: GetRolePolicy
	Iam:g etSAMLProvider
	saya: GetServerCertificate
	saya: GetServiceLastAccessedDetails
	saya: GetServiceLastAccessedDetailsWithEntities
	saya: GetServiceLinkedRoleDeletionStatus
	Iam:G Kunci etSSHPublic
	saya: GetUser
	saya: GetUserPolicy
	saya: ListAccessKeys
	saya: ListAccountAliases

Awalan layanan	Tindakan
	saya: ListAttachedGroupPolicies
	saya: ListAttachedRolePolicies
	saya: ListAttachedUserPolicies
	saya: ListCloudFrontPublicKeys
	saya: ListEntitiesForPolicy
	saya: ListGroupPolicies
	saya: ListGroups
	saya: ListGroupsForUser
	saya: ListInstanceProfiles
	saya: ListInstanceProfilesForRole
	Iam:L istMFADevices
	saya: ListOpen IDConnectProviders
	saya: ListPolicies
	saya: ListPoliciesGrantingServiceAccess
	saya: ListPolicyVersions
	saya: ListRolePolicies
	saya: ListRoles
	Iam:L istSAMLProviders
	saya: ListServerCertificates
	saya: ListServiceSpecificCredentials
	saya: ListSigningCertificates

Awalan layanan	Tindakan
	istSSHPublicIam:L Kunci
	Iam:L istSTSRegional EndpointsStatus
	saya: ListUserPolicies
	saya: ListUsers
	saya: ListVirtual MFADevices
	saya: PutGroupPolicy
	saya: PutRolePermissionsBoundary
	saya: PutRolePolicy
	saya: PutUserPermissionsBoundary
	saya: PutUserPolicy
	saya: RemoveClient IDFromOpenIDConnectProvider
	saya: RemoveRoleFromInstanceProfile
	saya: RemoveUserFromGroup
	saya: ResetServiceSpecificCredential
	Iam:R esyncMFADevice
	saya: SetDefaultPolicyVersion
	saya: SetSecurityTokenServicePreferences
	IAM:S etSTSRegional EndpointStatus
	saya: SimulateCustomPolicy
	saya: SimulatePrincipalPolicy
	saya: UpdateAccessKey

Awalan layanan	Tindakan
	saya: UpdateAccountEmailAddress
	saya: UpdateAccountName
	saya: UpdateAccountPasswordPolicy
	saya: UpdateAssumeRolePolicy
	saya: UpdateCloudFrontPublicKey
	saya: UpdateGroup
	saya: UpdateLoginProfile
	saya: UpdateOpen IDConnectProviderThumbprint
	saya: UpdateRole
	saya: UpdateRoleDescription
	Iam:U pdateSAMLProvider
	saya: UpdateServerCertificate
	saya: UpdateServiceSpecificCredential
	saya: UpdateSigningCertificate
	Iam:U Kunci pdateSSHPublic
	saya: UpdateUser
	saya: UploadCloudFrontPublicKey
	saya: UploadServerCertificate
	saya: UploadSigningCertificate
	Iam:U Kunci ploadSSHPublic

Awalan layanan	Tindakan
toko identitas	toko identitas: CreateGroup toko identitas: CreateGroupMembership toko identitas: CreateUser toko identitas: DeleteGroup toko identitas: DeleteGroupMembership toko identitas: DeleteUser toko identitas: DescribeGroup toko identitas: DescribeGroupMembership toko identitas: DescribeUser toko identitas: GetGroupId toko identitas: GetGroupMembershipId toko identitas: GetUserId toko identitas: IsMemberInGroups toko identitas: ListGroupMemberships toko identitas: ListGroupMembershipsForMember toko identitas: ListGroups toko identitas: ListUsers toko identitas: UpdateGroup toko identitas: UpdateUser

Awalan layanan	Tindakan
imagebuilder	imagebuilder: CancelImageCreation
	imagebuilder: CancelLifecycleExecution
	imagebuilder: CreateComponent
	imagebuilder: CreateContainerRecipe
	imagebuilder: CreateDistributionConfiguration
	imagebuilder: CreateImage
	imagebuilder: CreateImagePipeline
	imagebuilder: CreateImageRecipe
	imagebuilder: CreateInfrastructureConfiguration
	imagebuilder: CreateLifecyclePolicy
	imagebuilder: CreateWorkflow
	imagebuilder: DeleteComponent
	imagebuilder: DeleteContainerRecipe
	imagebuilder: DeleteDistributionConfiguration
	imagebuilder: DeleteImage
	imagebuilder: DeleteImagePipeline
	imagebuilder: DeleteImageRecipe
	imagebuilder: DeleteInfrastructureConfiguration
	imagebuilder: DeleteLifecyclePolicy
	imagebuilder: DeleteWorkflow
	imagebuilder: GetComponentPolicy

Awalan layanan	Tindakan
	imagebuilder: GetContainerRecipePolicy
	imagebuilder: GetImagePolicy
	imagebuilder: GetImageRecipePolicy
	imagebuilder: GetLifecycleExecution
	imagebuilder: GetLifecyclePolicy
	imagebuilder: GetWorkflowExecution
	imagebuilder: GetWorkflowStepExecution
	imagebuilder: ImportComponent
	imagebuilder: ImportVmImage
	imagebuilder: ListComponentBuildVersions
	imagebuilder: ListComponents
	imagebuilder: ListContainerRecipes
	imagebuilder: ListDistributionConfigurations
	imagebuilder: ListImageBuildVersions
	imagebuilder: ListImagePackages
	imagebuilder: ListImagePipelineImages
	imagebuilder: ListImagePipelines
	imagebuilder: ListImageRecipes
	imagebuilder: ListImages
	imagebuilder: ListImageScanFindingAggregations
	imagebuilder: ListImageScanFindings

Awalan layanan	Tindakan
	imagebuilder: ListInfrastructureConfigurations
	imagebuilder: ListLifecycleExecutionResources
	imagebuilder: ListLifecycleExecutions
	imagebuilder: ListLifecyclePolicies
	imagebuilder: ListWaitingWorkflowSteps
	imagebuilder: ListWorkflowExecutions
	imagebuilder: ListWorkflows
	imagebuilder: ListWorkflowStepExecutions
	imagebuilder: PutComponentPolicy
	imagebuilder: PutContainerRecipePolicy
	imagebuilder: PutImagePolicy
	imagebuilder: PutImageRecipePolicy
	imagebuilder: SendWorkflowStepAction
	imagebuilder: StartImagePipelineExecution
	imagebuilder: StartResourceStateUpdate
	imagebuilder: UpdateDistributionConfiguration
	imagebuilder: UpdateImagePipeline
	imagebuilder: UpdateInfrastructureConfiguration

Awalan layanan	Tindakan
inspektur	inspektur: AddAttributesToFindings inspektur: CreateAssessmentTarget inspektur: CreateAssessmentTemplate inspektur: CreateExclusionsPreview inspektur: CreateResourceGroup inspektur: DeleteAssessmentRun inspektur: DeleteAssessmentTarget inspektur: DeleteAssessmentTemplate inspektur: DescribeAssessmentRuns inspektur: DescribeAssessmentTargets inspektur: DescribeAssessmentTemplates inspektur: DescribeCrossAccountAccessRole inspektur: DescribeExclusions inspektur: DescribeFindings inspektur: DescribeResourceGroups inspektur: DescribeRulesPackages inspektur: GetAssessmentReport inspektur: GetExclusionsPreview inspektur: GetTelemetryMetadata inspektur: ListAssessmentRunAgents inspektur: ListAssessmentRuns

Awalan layanan	Tindakan
	inspektur: ListAssessmentTargets inspektur: ListAssessmentTemplates inspektur: ListEventSubscriptions inspektur: ListExclusions inspektur: ListFindings inspektur: ListRulesPackages inspektur: PreviewAgents inspektur: RegisterCrossAccountAccessRole inspektur: RemoveAttributesFromFindings inspektur: StartAssessmentRun inspektur: StopAssessmentRun inspektur: SubscribeToEvent inspektur: UnsubscribeFromEvent inspektur: UpdateAssessmentTarget

Awalan layanan	Tindakan
inspektor2	inspektor2: AssociateMember
	inspektor2: BatchGetAccountStatus
	inspektor2: BatchGetCodeSnippet
	inspektor2: BatchGetFindingDetails
	inspektor2: BatchGetFreeTrialInfo
	inspektor2:2 BatchGetMemberEc DeepInspectionStatus
	inspektor2:2 BatchUpdateMemberEc DeepInspectionStatus
	inspektor2: CancelFindingsReport
	inspektor2: CancelSbomExport
	inspektor2: CreateCisScanConfiguration
	inspektor2: CreateFilter
	inspektor2: CreateFindingsReport
	inspektor2: CreateSbomExport
	inspektor2: DeleteCisScanConfiguration
	inspektor2: DeleteFilter
	inspektor2: DescribeOrganizationConfiguration
	Inspector2: Nonaktifkan
	inspektor2: DisableDelegatedAdminAccount
	inspektor2: DisassociateMember
	Inspector2: aktifkan
	inspektor2: EnableDelegatedAdminAccount

Awalan layanan	Tindakan
	inspektor2: GetCisScanReport
	inspektor2: GetCisScanResultDetails
	inspektor2: GetConfiguration
	inspektor2: GetDelegatedAdminAccount
	inspektor2:2 GetEc DeepInspectionConfiguration
	inspektor2: GetEncryptionKey
	inspektor2: GetFindingsReportStatus
	inspektor2: GetMember
	inspektor2: GetSbomExport
	inspektor2: ListAccountPermissions
	inspektor2: ListCisScanConfigurations
	inspektor2: ListCisScanResultsAggregatedByChecks
	inspektor2: ListCisScanResultsAggregatedByTargetResource
	inspektor2: ListCisScans
	inspektor2: ListCoverage
	inspektor2: ListCoverageStatistics
	inspektor2: ListDelegatedAdminAccounts
	inspektor2: ListFilters
	inspektor2: ListFindingAggregations
	inspektor2: ListFindings
	inspektor2: ListMembers

Awalan layanan	Tindakan
	inspektor2: ListUsageTotals
	inspektor2: ResetEncryptionKey
	inspektor2: SearchVulnerabilities
	inspektor2: SendCisSessionHealth
	inspektor2: SendCisSessionTelemetry
	inspektor2: StartCisSession
	inspektor2: StopCisSession
	inspektor2: UpdateCisScanConfiguration
	inspektor2: UpdateConfiguration
	inspektor2:2 UpdateEc DeepInspectionConfiguration
	inspektor2: UpdateEncryptionKey
	inspektor2: UpdateFilter
	inspektor2: UpdateOrganizationConfiguration
	inspektor2:2 UpdateOrgEc DeepInspectionConfiguration

Awalan layanan	Tindakan
iot	IOT: AcceptCertificateTransfer
	IOT: AddThingToBillingGroup
	IOT: AddThingToThingGroup
	IOT: AssociateTargetsWithJob
	IOT: AttachPolicy
	IOT: AttachPrincipalPolicy
	IOT: AttachSecurityProfile
	IOT: AttachThingPrincipal
	IOT: CancelAuditMitigationActionsTask
	IOT: CancelAuditTask
	IOT: CancelCertificateTransfer
	IOT: CancelDetectMitigationActionsTask
	IOT: CancelJob
	IOT: CancelJobExecution
	IOT: ClearDefaultAuthorizer
	IOT: ConfirmTopicRuleDestination
	IOT: CreateAuditSuppression
	IOT: CreateAuthorizer
	IOT: CreateBillingGroup
	IOT: CreateCertificateFromCsr
	IOT: CreateCertificateProvider

Awalan layanan	Tindakan
	IOT: CreateCustomMetric IOT: CreateDimension IOT: CreateDomainConfiguration IOT: CreateDynamicThingGroup IOT: CreateFleetMetric IOT: CreateJob IOT: CreateJobTemplate IOT: CreateKeysAndCertificate IOT: CreateMitigationAction IoT: C reateOTAUpdate IOT: CreatePackage IOT: CreatePackageVersion IOT: CreatePolicy IOT: CreatePolicyVersion IOT: CreateProvisioningClaim IOT: CreateProvisioningTemplate IOT: CreateProvisioningTemplateVersion IOT: CreateRoleAlias IOT: CreateScheduledAudit IOT: CreateSecurityProfile IOT: CreateStream

Awalan layanan	Tindakan
	IOT: CreateThing IOT: CreateThingGroup IOT: CreateThingType IOT: CreateTopicRule IOT: CreateTopicRuleDestination IOT: DeleteAccountAuditConfiguration IOT: DeleteAuditSuppression IOT: DeleteAuthorizer IOT: DeleteBillingGroup IoT: D eleteCACertificate IOT: DeleteCertificate IOT: DeleteCertificateProvider IOT: DeleteCustomMetric IOT: DeleteDimension IOT: DeleteDomainConfiguration IOT: DeleteDynamicThingGroup IOT: DeleteFleetMetric IOT: DeleteJob IOT: DeleteJobExecution IOT: DeleteJobTemplate IOT: DeleteMitigationAction

Awalan layanan	Tindakan
	IoT: D eleteOTAUpdate
	IOT: DeletePackage
	IOT: DeletePackageVersion
	IOT: DeletePolicy
	IOT: DeletePolicyVersion
	IOT: DeleteProvisioningTemplate
	IOT: DeleteProvisioningTemplateVersion
	IOT: DeleteRegistrationCode
	IOT: DeleteRoleAlias
	IOT: DeleteScheduledAudit
	IOT: DeleteSecurityProfile
	IOT: DeleteStream
	IOT: DeleteThing
	IOT: DeleteThingGroup
	IOT: DeleteThingType
	IOT: DeleteTopicRule
	IOT: DeleteTopicRuleDestination
	IoT: DeleteV2 LoggingLevel
	IOT: DeprecateThingType
	IOT: DescribeAccountAuditConfiguration
	IOT: DescribeAuditFinding

Awalan layanan	Tindakan
	IOT: DescribeAuditMitigationActionsTask IOT: DescribeAuditSuppression IOT: DescribeAuditTask IOT: DescribeAuthorizer IOT: DescribeBillingGroup IoT: D escribeCACertificate IOT: DescribeCertificate IOT: DescribeCertificateProvider IOT: DescribeCustomMetric IOT: DescribeDefaultAuthorizer IOT: DescribeDetectMitigationActionsTask IOT: DescribeDimension IOT: DescribeDomainConfiguration IOT: DescribeEndpoint IOT: DescribeEventConfigurations IOT: DescribeFleetMetric IOT: DescribeIndex IOT: DescribeJob IOT: DescribeJobExecution IOT: DescribeJobTemplate IOT: DescribeManagedJobTemplate

Awalan layanan	Tindakan
	IOT: DescribeMitigationAction IOT: DescribeProvisioningTemplate IOT: DescribeProvisioningTemplateVersion IOT: DescribeRoleAlias IOT: DescribeScheduledAudit IOT: DescribeSecurityProfile IOT: DescribeStream IOT: DescribeThing IOT: DescribeThingGroup IOT: DescribeThingRegistrationTask IOT: DescribeThingType IOT: DetachPolicy IOT: DetachPrincipalPolicy IOT: DetachSecurityProfile IOT: DetachThingPrincipal IOT: DisableTopicRule IOT: EnableTopicRule IOT: GetBehaviorModelTrainingSummaries IOT: GetBucketsAggregation IOT: GetCardinality IOT: GetEffectivePolicies

Awalan layanan	Tindakan
	IOT: GetJobDocument
	IOT: GetLoggingOptions
	IoT: G etOTAUpdate
	IOT: GetPackage
	IOT: GetPackageConfiguration
	IOT: GetPackageVersion
	IOT: GetPercentiles
	IOT: GetPolicy
	IOT: GetPolicyVersion
	IOT: GetRegistrationCode
	IOT: GetStatistics
	IOT: GetTopicRule
	IOT: GetTopicRuleDestination
	IoT: GetV2 LoggingOptions
	IOT: ListActiveViolations
	IOT: ListAttachedPolicies
	IOT: ListAuditFindings
	IOT: ListAuditMitigationActionsExecutions
	IOT: ListAuditMitigationActionsTasks
	IOT: ListAuditSuppressions
	IOT: ListAuditTasks

Awalan layanan	Tindakan
	IOT: ListAuthorizers IOT: ListBillingGroups IoT: L istCACertificates IOT: ListCertificateProviders IOT: ListCertificates IoT: CA ListCertificatesBy IOT: ListCustomMetrics IOT: ListDetectMitigationActionsExecutions IOT: ListDetectMitigationActionsTasks IOT: ListDimensions IOT: ListDomainConfigurations IOT: ListFleetMetrics IOT: ListIndices IOT: ListJobExecutionsForJob IOT: ListJobExecutionsForThing IOT: ListJobs IOT: ListJobTemplates IOT: ListManagedJobTemplates IOT: ListMetricValues IOT: ListMitigationActions IoT: L istOTAUpdates

Awalan layanan	Tindakan
	IOT: ListOutgoingCertificates IOT: ListPackages IOT: ListPackageVersions IOT: ListPolicies IOT: ListPolicyPrincipals IOT: ListPolicyVersions IOT: ListPrincipalPolicies IOT: ListPrincipalThings IOT: ListProvisioningTemplates IOT: ListProvisioningTemplateVersions IOT: ListRelatedResourcesForAuditFinding IOT: ListRoleAliases IOT: ListScheduledAudits IOT: ListSecurityProfiles IOT: ListSecurityProfilesForTarget IOT: ListStreams IOT: ListTargetsForPolicy IOT: ListTargetsForSecurityProfile IOT: ListThingGroups IOT: ListThingGroupsForThing IOT: ListThingPrincipals

Awalan layanan	Tindakan
	IOT: ListThingRegistrationTaskReports
	IOT: ListThingRegistrationTasks
	IOT: ListThings
	IOT: ListThingsInBillingGroup
	IOT: ListThingsInThingGroup
	IOT: ListThingTypes
	IOT: ListTopicRuleDestinations
	IOT: ListTopicRules
	IoT: Listv2 LoggingLevels
	IOT: ListViolationEvents
	IOT: PutVerificationStateOnViolation
	IoT: R egisterCACertificate
	IOT: RegisterCertificate
	IoT: CA RegisterCertificateWithout
	IOT: RegisterThing
	IOT: RejectCertificateTransfer
	IOT: RemoveThingFromBillingGroup
	IOT: RemoveThingFromThingGroup
	IOT: ReplaceTopicRule
	IOT: SearchIndex
	IOT: SetDefaultAuthorizer

Awalan layanan	Tindakan
	IOT: SetDefaultPolicyVersion
	IOT: SetLoggingOptions
	IoT: Setv2 LoggingLevel
	IoT: Setv2 LoggingOptions
	IOT: StartAuditMitigationActionsTask
	IOT: StartDetectMitigationActionsTask
	IOT: StartOnDemandAuditTask
	IOT: StartThingRegistrationTask
	IOT: StopThingRegistrationTask
	IOT: TestAuthorization
	IOT: TestInvokeAuthorizer
	IOT: TransferCertificate
	IOT: UpdateAccountAuditConfiguration
	IOT: UpdateAuditSuppression
	IOT: UpdateAuthorizer
	IOT: UpdateBillingGroup
	IoT: U pdateCACertificate
	IOT: UpdateCertificate
	IOT: UpdateCertificateProvider
	IOT: UpdateCustomMetric
	IOT: UpdateDimension

Awalan layanan	Tindakan
	IOT: UpdateDomainConfiguration IOT: UpdateDynamicThingGroup IOT: UpdateEventConfigurations IOT: UpdateFleetMetric IOT: UpdateIndexingConfiguration IOT: UpdateJob IOT: UpdateMitigationAction IOT: UpdatePackage IOT: UpdatePackageConfiguration IOT: UpdatePackageVersion IOT: UpdateProvisioningTemplate IOT: UpdateRoleAlias IOT: UpdateScheduledAudit IOT: UpdateSecurityProfile IOT: UpdateStream IOT: UpdateThing IOT: UpdateThingGroup IOT: UpdateThingGroupsForThing IOT: UpdateTopicRuleDestination IOT: ValidateSecurityProfileBehaviors

Awalan layanan	Tindakan
iotanalitik	iotanalitik: CancelPipelineReprocessing
	iotanalitik: CreateChannel
	iotanalitik: CreateDataset
	iotanalitik: CreateDatasetContent
	iotanalitik: CreateDatastore
	iotanalitik: CreatePipeline
	iotanalitik: DeleteChannel
	iotanalitik: DeleteDataset
	iotanalitik: DeleteDatasetContent
	iotanalitik: DeleteDatastore
	iotanalitik: DeletePipeline
	iotanalitik: DescribeChannel
	iotanalitik: DescribeDataset
	iotanalitik: DescribeDatastore
	iotanalitik: DescribeLoggingOptions
	iotanalitik: DescribePipeline
	iotanalitik: GetDatasetContent
	iotanalitik: ListChannels
	iotanalitik: ListDatasetContents
	iotanalitik: ListDatasets
	iotanalitik: ListDatastores

Awalan layanan	Tindakan
	iotanalitik: ListPipelines iotanalitik: PutLoggingOptions iotanalitik: RunPipelineActivity iotanalitik: SampleChannelData iotanalitik: StartPipelineReprocessing iotanalitik: UpdateChannel iotanalitik: UpdateDataset iotanalitik: UpdateDatastore iotanalitik: UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor: CreateSuiteDefinition iotdeviceadvisor: DeleteSuiteDefinition iotdeviceadvisor: GetEndpoint iotdeviceadvisor: GetSuiteDefinition iotdeviceadvisor: GetSuiteRun iotdeviceadvisor: GetSuiteRunReport iotdeviceadvisor: ListSuiteDefinitions iotdeviceadvisor: ListSuiteRuns iotdeviceadvisor: StartSuiteRun iotdeviceadvisor: StopSuiteRun iotdeviceadvisor: UpdateSuiteDefinition

Awalan layanan	Tindakan
iotevents	iotevents: BatchAcknowledgeAlarm
	iotevents: BatchDeleteDetector
	iotevents: BatchDisableAlarm
	iotevents: BatchEnableAlarm
	iotevents: BatchResetAlarm
	iotevents: BatchSnoozeAlarm
	iotevents: BatchUpdateDetector
	iotevents: CreateAlarmModel
	iotevents: CreateDetectorModel
	iotevents: CreateInput
	iotevents: DeleteAlarmModel
	iotevents: DeleteDetectorModel
	iotevents: DeleteInput
	iotevents: DescribeAlarm
	iotevents: DescribeAlarmModel
	iotevents: DescribeDetector
	iotevents: DescribeDetectorModel
	iotevents: DescribeDetectorModelAnalysis
	iotevents: DescribeInput
	iotevents: DescribeLoggingOptions
	iotevents: GetDetectorModelAnalysisResults

Awalan layanan	Tindakan
	iotevents: ListAlarmModels iotevents: ListAlarmModelVersions iotevents: ListAlarms iotevents: ListDetectorModels iotevents: ListDetectorModelVersions iotevents: ListDetectors iotevents: ListInputRoutings iotevents: ListInputs iotevents: PutLoggingOptions iotevents: StartDetectorModelAnalysis iotevents: UpdateAlarmModel iotevents: UpdateDetectorModel iotevents: UpdateInput
iotfleethub	iotfleethub: CreateApplication iotfleethub: DeleteApplication iotfleethub: DescribeApplication iotfleethub: ListApplications iotfleethub: UpdateApplication

Awalan layanan	Tindakan
iotsitewise	iotsitewise: AssociateAssets
	iotsitewise: AssociateTimeSeriesToAssetProperty
	iotsitewise: BatchAssociateProjectAssets
	iotsitewise: BatchDisassociateProjectAssets
	iotsitewise: BatchGetAssetPropertyValue
	iotsitewise: BatchGetAssetPropertyValueHistory
	iotsitewise: BatchPutAssetPropertyValue
	iotsitewise: CreateAccessPolicy
	iotsitewise: CreateAsset
	iotsitewise: CreateAssetModel
	iotsitewise: CreateAssetModelCompositeModel
	iotsitewise: CreateBulkImportJob
	iotsitewise: CreateDashboard
	iotsitewise: CreateGateway
	iotsitewise: CreatePortal
	iotsitewise: CreateProject
	iotsitewise: DeleteAccessPolicy
	iotsitewise: DeleteAsset
	iotsitewise: DeleteAssetModel
	iotsitewise: DeleteAssetModelCompositeModel
	iotsitewise: DeleteDashboard

Awalan layanan	Tindakan
	iotsitewise: DeleteGateway
	iotsitewise: DeletePortal
	iotsitewise: DeleteProject
	iotsitewise: DeleteTimeSeries
	iotsitewise: DescribeAccessPolicy
	iotsitewise: DescribeAsset
	iotsitewise: DescribeAssetCompositeModel
	iotsitewise: DescribeAssetModel
	iotsitewise: DescribeAssetModelCompositeModel
	iotsitewise: DescribeAssetProperty
	iotsitewise: DescribeBulkImportJob
	iotsitewise: DescribeDashboard
	iotsitewise: DescribeDefaultEncryptionConfiguration
	iotsitewise: DescribeGateway
	iotsitewise: DescribeGatewayCapabilityConfiguration
	iotsitewise: DescribeLoggingOptions
	iotsitewise: DescribePortal
	iotsitewise: DescribeProject
	iotsitewise: DescribeStorageConfiguration
	iotsitewise: DescribeTimeSeries
	iotsitewise: DisassociateAssets

Awalan layanan	Tindakan
	iotsitewise: DisassociateTimeSeriesFromAssetProperty
	iotsitewise: ExecuteAction
	iotsitewise: ExecuteQuery
	iotsitewise: ListAccessPolicies
	iotsitewise: ListActions
	iotsitewise: ListAssetModelCompositeModels
	iotsitewise: ListAssetModelProperties
	iotsitewise: ListAssetModels
	iotsitewise: ListAssetProperties
	iotsitewise: ListAssetRelationships
	iotsitewise: ListAssets
	iotsitewise: ListAssociatedAssets
	iotsitewise: ListBulkImportJobs
	iotsitewise: ListCompositionRelationships
	iotsitewise: ListDashboards
	iotsitewise: ListGateways
	iotsitewise: ListPortals
	iotsitewise: ListProjectAssets
	iotsitewise: ListProjects
	iotsitewise: ListTimeSeries
	iotsitewise: PutDefaultEncryptionConfiguration

Awalan layanan	Tindakan
	iotsitewise: PutLoggingOptions
	iotsitewise: PutStorageConfiguration
	iotsitewise: UpdateAccessPolicy
	iotsitewise: UpdateAsset
	iotsitewise: UpdateAssetModel
	iotsitewise: UpdateAssetModelCompositeModel
	iotsitewise: UpdateAssetProperty
	iotsitewise: UpdateDashboard
	iotsitewise: UpdateGateway
	iotsitewise: UpdateGatewayCapabilityConfiguration
	iotsitewise: UpdatePortal
	iotsitewise: UpdateProject

Awalan layanan	Tindakan
pembuat iottwinmaker	pembuat iottwinmaker: CancelMetadataTransferJob pembuat iottwinmaker: CreateComponentType pembuat iottwinmaker: CreateEntity pembuat iottwinmaker: CreateMetadataTransferJob pembuat iottwinmaker: CreateScene pembuat iottwinmaker: CreateSyncJob pembuat iottwinmaker: CreateWorkspace pembuat iottwinmaker: DeleteComponentType pembuat iottwinmaker: DeleteEntity pembuat iottwinmaker: DeleteScene pembuat iottwinmaker: DeleteSyncJob pembuat iottwinmaker: DeleteWorkspace pembuat iottwinmaker: ExecuteQuery pembuat iottwinmaker: GetMetadataTransferJob pembuat iottwinmaker: GetPricingPlan pembuat iottwinmaker: GetScene pembuat iottwinmaker: GetSyncJob pembuat iottwinmaker: ListComponents pembuat iottwinmaker: ListComponentTypes pembuat iottwinmaker: ListEntities pembuat iottwinmaker: ListMetadataTransferJobs

Awalan layanan	Tindakan
	pembuat iottwinmaker: ListProperties pembuat iottwinmaker: ListScenes pembuat iottwinmaker: ListSyncJobs pembuat iottwinmaker: ListSyncResources pembuat iottwinmaker: ListWorkspaces pembuat iottwinmaker: UpdateComponentType pembuat iottwinmaker: UpdateEntity pembuat iottwinmaker: UpdatePricingPlan pembuat iottwinmaker: UpdateScene pembuat iottwinmaker: UpdateWorkspace

Awalan layanan	Tindakan
iotwireless	iotwireless: AssociateAwsAccountWithPartnerAccount
	iotwireless: AssociateMulticastGroupWithFuotaTask
	iotwireless: AssociateWirelessDeviceWithFuotaTask
	iotwireless: AssociateWirelessDeviceWithMulticastGroup
	iotwireless: AssociateWirelessDeviceWithThing
	iotwireless: AssociateWirelessGatewayWithCertificate
	iotwireless: AssociateWirelessGatewayWithThing
	iotwireless: CancelMulticastGroupSession
	iotwireless: CreateDestination
	iotwireless: CreateDeviceProfile
	iotwireless: CreateFuotaTask
	iotwireless: CreateMulticastGroup
	iotwireless: CreateNetworkAnalyzerConfiguration
	iotwireless: CreateServiceProfile
	iotwireless: CreateWirelessDevice
	iotwireless: CreateWirelessGateway
	iotwireless: CreateWirelessGatewayTask
	iotwireless: CreateWirelessGatewayTaskDefinition
	iotwireless: DeleteDestination
	iotwireless: DeleteDeviceProfile
	iotwireless: DeleteFuotaTask

Awalan layanan	Tindakan
	iotwireless: DeleteMulticastGroup
	iotwireless: DeleteNetworkAnalyzerConfiguration
	iotwireless: DeleteQueuedMessages
	iotwireless: DeleteServiceProfile
	iotwireless: DeleteWirelessDevice
	iotwireless: DeleteWirelessDeviceImportTask
	iotwireless: DeleteWirelessGateway
	iotwireless: DeleteWirelessGatewayTask
	iotwireless: DeleteWirelessGatewayTaskDefinition
	iotwireless: DeregisterWirelessDevice
	iotwireless: DisassociateAwsAccountFromPartnerAccount
	iotwireless: DisassociateMulticastGroupFromFuotaTask
	iotwireless: DisassociateWirelessDeviceFromFuotaTask
	iotwireless: DisassociateWirelessDeviceFromMulticastGroup
	iotwireless: DisassociateWirelessDeviceFromThing
	iotwireless: DisassociateWirelessGatewayFromCertificate
	iotwireless: DisassociateWirelessGatewayFromThing
	iotwireless: GetDestination
	iotwireless: GetDeviceProfile
	iotwireless: GetEventConfigurationByResourceTypes
	iotwireless: GetFuotaTask

Awalan layanan	Tindakan
	iotwireless: GetLogLevelsByResourceTypes
	iotwireless: GetMetricConfiguration
	iotwireless: GetMetrics
	iotwireless: GetMulticastGroup
	iotwireless: GetMulticastGroupSession
	iotwireless: GetNetworkAnalyzerConfiguration
	iotwireless: GetPartnerAccount
	iotwireless: GetPosition
	iotwireless: GetPositionConfiguration
	iotwireless: GetPositionEstimate
	iotwireless: GetResourceEventConfiguration
	iotwireless: GetResourceLogLevel
	iotwireless: GetResourcePosition
	iotwireless: GetServiceEndpoint
	iotwireless: GetServiceProfile
	iotwireless: GetWirelessDevice
	iotwireless: GetWirelessDeviceImportTask
	iotwireless: GetWirelessDeviceStatistics
	iotwireless: GetWirelessGateway
	iotwireless: GetWirelessGatewayCertificate
	iotwireless: GetWirelessGatewayFirmwareInformation

Awalan layanan	Tindakan
	iotwireless: GetWirelessGatewayStatistics
	iotwireless: GetWirelessGatewayTask
	iotwireless: GetWirelessGatewayTaskDefinition
	iotwireless: ListDestinations
	iotwireless: ListDeviceProfiles
	iotwireless: ListDevicesForWirelessDeviceImportTask
	iotwireless: ListEventConfigurations
	iotwireless: ListFuotaTasks
	iotwireless: ListMulticastGroups
	iotwireless: ListMulticastGroupsByFuotaTask
	iotwireless: ListNetworkAnalyzerConfigurations
	iotwireless: ListPartnerAccounts
	iotwireless: ListPositionConfigurations
	iotwireless: ListQueuedMessages
	iotwireless: ListServiceProfiles
	iotwireless: ListWirelessDeviceImportTasks
	iotwireless: ListWirelessDevices
	iotwireless: ListWirelessGateways
	iotwireless: ListWirelessGatewayTaskDefinitions
	iotwireless: PutPositionConfiguration
	iotwireless: PutResourceLogLevel

Awalan layanan	Tindakan
	iotwireless: ResetAllResourceLogLevels
	iotwireless: ResetResourceLogLevel
	iotwireless: SendDataToMulticastGroup
	iotwireless: SendDataToWirelessDevice
	iotwireless: StartBulkAssociateWirelessDeviceWithMulticastGroup
	iotwireless: StartBulkDisassociateWirelessDeviceFromMulticastGroup
	iotwireless: StartFuotaTask
	iotwireless: StartMulticastGroupSession
	iotwireless: StartNetworkAnalyzerStream
	iotwireless: StartSingleWirelessDeviceImportTask
	iotwireless: StartWirelessDeviceImportTask
	iotwireless: TestWirelessDevice
	iotwireless: UpdateDestination
	iotwireless: UpdateEventConfigurationByResourceTypes
	iotwireless: UpdateFuotaTask
	iotwireless: UpdateLogLevelsByResourceTypes
	iotwireless: UpdateMetricConfiguration
	iotwireless: UpdateMulticastGroup
	iotwireless: UpdateNetworkAnalyzerConfiguration
	iotwireless: UpdatePartnerAccount

Awalan layanan	Tindakan
	iotwireless: UpdatePosition iotwireless: UpdateResourceEventConfiguration iotwireless: UpdateResourcePosition iotwireless: UpdateWirelessDevice iotwireless: UpdateWirelessDeviceImportTask iotwireless: UpdateWirelessGateway

Awalan layanan	Tindakan
ivs	ivs: BatchGetChannel ivs: BatchGetStreamKey ivs: BatchStartViewerSessionRevocation ivs: CreateChannel ivs: CreateEncoderConfiguration ivs: CreateParticipantToken ivs: CreatePlaybackRestrictionPolicy ivs: CreateRecordingConfiguration ivs: CreateStorageConfiguration ivs: CreateStreamKey ivs: DeleteChannel ivs: DeleteEncoderConfiguration ivs: DeletePlaybackKeyPair ivs: DeletePlaybackRestrictionPolicy ivs: DeletePublicKey ivs: DeleteRecordingConfiguration ivs: DeleteStorageConfiguration ivs: DeleteStreamKey ivs: DisconnectParticipant ivs: GetChannel ivs: GetComposition

Awalan layanan	Tindakan
	ivs: GetEncoderConfiguration ivs: GetParticipant ivs: GetPlaybackKeyPair ivs: GetPlaybackRestrictionPolicy ivs: GetPublicKey ivs: GetRecordingConfiguration ivs: GetStorageConfiguration ivs: GetStream ivs: GetStreamKey ivs: GetStreamSession ivs: ImportPlaybackKeyPair ivs: ImportPublicKey ivs: ListChannels ivs: ListCompositions ivs: ListEncoderConfigurations ivs: ListParticipantEvents ivs: ListParticipants ivs: ListPlaybackKeyPairs ivs: ListPlaybackRestrictionPolicies ivs: ListPublicKeys ivs: ListRecordingConfigurations

Awalan layanan	Tindakan
	ivs: ListStorageConfigurations ivs: ListStreamKeys ivs: ListStreams ivs: ListStreamSessions ivs: PutMetadata ivs: StartComposition ivs: StartViewerSessionRevocation ivs: StopComposition ivs: StopStream ivs: UpdateChannel ivs: UpdatePlaybackRestrictionPolicy

Awalan layanan	Tindakan
ivschat	ivschat: CreateChatToken ivschat: CreateLoggingConfiguration ivschat: CreateRoom ivschat: DeleteLoggingConfiguration ivschat: DeleteMessage ivschat: DeleteRoom ivschat: DisconnectUser ivschat: GetLoggingConfiguration ivschat: GetRoom ivschat: ListLoggingConfigurations ivschat: ListRooms ivschat: SendEvent ivschat: UpdateLoggingConfiguration ivschat: UpdateRoom

Awalan layanan	Tindakan
kafka	kafka: BatchAssociateScramSecret kafka: BatchDisassociateScramSecret kafka: CreateCluster kafka: CreateCluster V2 kafka: CreateConfiguration kafka: CreateReplicator kafka: CreateVpcConnection kafka: DeleteCluster kafka: DeleteClusterPolicy kafka: DeleteConfiguration kafka: DeleteReplicator kafka: DeleteVpcConnection kafka: DescribeCluster kafka: DescribeClusterOperation kafka: DescribeClusterOperation V2 kafka: DescribeCluster V2 kafka: DescribeConfiguration kafka: DescribeConfigurationRevision kafka: DescribeVpcConnection kafka: GetBootstrapBrokers kafka: GetClusterPolicy

Awalan layanan	Tindakan
	kafka: GetCompatibleKafkaVersions kafka: ListClientVpcConnections kafka: ListClusterOperations kafka: ListClusterOperations V2 kafka: ListClusters kafka: ListClusters V2 kafka: ListConfigurationRevisions kafka: ListConfigurations kafka: ListKafkaVersions kafka: ListNodes kafka: ListReplicators kafka: ListScramSecrets kafka: ListVpcConnections kafka: PutClusterPolicy kafka: RebootBroker kafka: RejectClientVpcConnection kafka: UpdateBrokerCount kafka: UpdateBrokerStorage kafka: UpdateBrokerType kafka: UpdateClusterConfiguration kafka: UpdateClusterKafkaVersion

Awalan layanan	Tindakan
	kafka: UpdateConfiguration kafka: UpdateConnectivity kafka: UpdateMonitoring kafka: UpdateReplicationInfo kafka: UpdateSecurity kafka: UpdateStorage
kafkaconnect	kafkaconnect: CreateConnector kafkaconnect: CreateCustomPlugin kafkaconnect: CreateWorkerConfiguration kafkaconnect: DeleteConnector kafkaconnect: DeleteCustomPlugin kafkaconnect: DeleteWorkerConfiguration kafkaconnect: DescribeConnector kafkaconnect: DescribeCustomPlugin kafkaconnect: DescribeWorkerConfiguration kafkaconnect: ListConnectors kafkaconnect: ListCustomPlugins kafkaconnect: ListWorkerConfigurations kafkaconnect: UpdateConnector

Awalan layanan	Tindakan
kendra	kendra: AssociateEntitiesToExperience kendra: AssociatePersonasToEntities kendra: BatchDeleteDocument kendra: BatchDeleteFeaturedResultsSet kendra: BatchGetDocumentStatus kendra: BatchPutDocument kendra: ClearQuerySuggestions kendra: CreateAccessControlConfiguration kendra: CreateDataSource kendra: CreateExperience kendra: CreateFaq kendra: CreateFeaturedResultsSet kendra: CreateIndex kendra: CreateQuerySuggestionsBlockList kendra: CreateThesaurus kendra: DeleteDataSource kendra: DeleteExperience kendra: DeleteFaq kendra: DeleteIndex kendra: DeletePrincipalMapping kendra: DeleteQuerySuggestionsBlockList

Awalan layanan	Tindakan
	kendra: DeleteThesaurus kendra: DescribeAccessControlConfiguration kendra: DescribeDataSource kendra: DescribeExperience kendra: DescribeFaq kendra: DescribeFeaturedResultsSet kendra: DescribeIndex kendra: DescribePrincipalMapping kendra: DescribeQuerySuggestionsBlockList kendra: DescribeQuerySuggestionsConfig kendra: DescribeThesaurus kendra: DisassociateEntitiesFromExperience kendra: DisassociatePersonasFromEntities kendra: GetQuerySuggestions kendra: GetSnapshots kendra: ListAccessControlConfigurations kendra: ListDataSources kendra: ListDataSourceSyncJobs kendra: ListEntityPersonas kendra: ListExperienceEntities kendra: ListExperiences

Awalan layanan	Tindakan
	kendra: ListFaqs kendra: ListFeaturedResultsSets kendra: ListGroupsOlderThanOrderingId kendra: ListIndices kendra: ListQuerySuggestionsBlockLists kendra: ListThesauri kendra: PutPrincipalMapping Kendra:Pertanyaan Kendra:ambil kendra: StartDataSourceSyncJob kendra: StopDataSourceSyncJob kendra: SubmitFeedback kendra: UpdateDataSource kendra: UpdateExperience kendra: UpdateFeaturedResultsSet kendra: UpdateIndex kendra: UpdateQuerySuggestionsBlockList kendra: UpdateQuerySuggestionsConfig kendra: UpdateThesaurus

Awalan layanan	Tindakan
kinesis	kinesis: CreateStream
	kinesis: DecreaseStreamRetentionPeriod
	kinesis: DeleteStream
	kinesis: DeregisterStreamConsumer
	kinesis: DescribeLimits
	kinesis: DescribeStream
	kinesis: DescribeStreamConsumer
	kinesis: DescribeStreamSummary
	kinesis: DisableEnhancedMonitoring
	kinesis: EnableEnhancedMonitoring
	kinesis: IncreaseStreamRetentionPeriod
	kinesis: ListShards
	kinesis: ListStreamConsumers
	kinesis: ListStreams
	kinesis: MergeShards
	kinesis: RegisterStreamConsumer
	kinesis: SplitShard
	kinesis: StartStreamEncryption
	kinesis: StopStreamEncryption
	kinesis: UpdateShardCount
	kinesis: UpdateStreamMode

Awalan layanan	Tindakan
kinesisanalitik	kinesisanalitik: AddApplicationCloudWatchLoggingOption kinesisanalitik: AddApplicationInput kinesisanalitik: AddApplicationInputProcessingConfiguration kinesisanalitik: AddApplicationOutput kinesisanalitik: AddApplicationReferenceDataSource kinesisanalitik: AddApplicationVpcConfiguration kinesisanalitik: CreateApplication kinesisanalitik: CreateApplicationPresignedUrl kinesisanalitik: CreateApplicationSnapshot kinesisanalitik: DeleteApplication kinesisanalitik: DeleteApplicationCloudWatchLoggingOption kinesisanalitik: DeleteApplicationInputProcessingConfiguration kinesisanalitik: DeleteApplicationOutput kinesisanalitik: DeleteApplicationReferenceDataSource kinesisanalitik: DeleteApplicationSnapshot kinesisanalitik: DeleteApplicationVpcConfiguration kinesisanalitik: DescribeApplication kinesisanalitik: DescribeApplicationOperation kinesisanalitik: DescribeApplicationSnapshot kinesisanalitik: DescribeApplicationVersion kinesisanalitik: DiscoverInputSchema

Awalan layanan	Tindakan
	kinesisanalitik: ListApplicationOperations kinesisanalitik: ListApplications kinesisanalitik: ListApplicationSnapshots kinesisanalitik: ListApplicationVersions kinesisanalitik: RollbackApplication kinesisanalitik: StartApplication kinesisanalitik: StopApplication kinesisanalitik: UpdateApplication kinesisanalitik: UpdateApplicationMaintenanceConfiguration

Awalan layanan	Tindakan
km	km: CancelKeyDeletion
	km: ConnectCustomKeyStore
	km: CreateAlias
	km: CreateCustomKeyStore
	km: CreateGrant
	km: CreateKey
	kms:Decrypt
	km: DeleteAlias
	km: DeleteCustomKeyStore
	km: DeleteImportedKeyMaterial
	km: DeriveSharedSecret
	km: DescribeCustomKeyStores
	km: DescribeKey
	km: DisableKey
	km: DisableKeyRotation
	km: DisconnectCustomKeyStore
	km: EnableKey
	km: EnableKeyRotation
	kms:Encrypt
	km: GenerateDataKey
	km: GenerateDataKeyPair

Awalan layanan	Tindakan
	km: GenerateDataKeyPairWithoutPlaintext
	km: GenerateDataKeyWithoutPlaintext
	km: GenerateMac
	km: GenerateRandom
	km: GetKeyPolicy
	km: GetKeyRotationStatus
	km: GetParametersForImport
	km: GetPublicKey
	km: ImportKeyMaterial
	km: ListAliases
	km: ListGrants
	km: ListKeyPolicies
	km: ListKeyRotations
	km: ListKeys
	km: ListRetirableGrants
	km: ReplicateKey
	km: RetireGrant
	km: RevokeGrant
	km: RotateKeyOnDemand
	km: ScheduleKeyDeletion
	KMS: Tanda

Awalan layanan	Tindakan
	km: UpdateAlias km: UpdateCustomKeyStore km: UpdateKeyDescription km: UpdatePrimaryRegion KMS: Verifikasi km: VerifyMac

Awalan layanan	Tindakan
lambda	lambda: AddLayerVersionPermission
	lambda: AddLayerVersionPermission
	lambda: AddPermission
	lambda: AddPermission
	lambda: AddPermission
	lambda: CreateAlias
	lambda: CreateAlias
	lambda: CreateCodeSigningConfig
	lambda: CreateEventSourceMapping
	lambda: CreateEventSourceMapping
	lambda: CreateFunction
	lambda: CreateFunction
	lambda: CreateFunctionUrlConfig
	lambda: DeleteAlias
	lambda: DeleteAlias
	lambda: DeleteCodeSigningConfig
	lambda: DeleteEventSourceMapping
	lambda: DeleteEventSourceMapping
	lambda: DeleteFunction
	lambda: DeleteFunction
	lambda: DeleteFunctionCodeSigningConfig

Awalan layanan	Tindakan
	lambda: DeleteFunctionConcurrency
	lambda: DeleteFunctionConcurrency
	lambda: DeleteFunctionEventInvokeConfig
	lambda: DeleteFunctionUrlConfig
	lambda: DeleteLayerVersion
	lambda: DeleteLayerVersion
	lambda: DeleteProvisionedConcurrencyConfig
	lambda: GetAccountSettings
	lambda: GetAccountSettings
	lambda: GetAlias
	lambda: GetAlias
	lambda: GetCodeSigningConfig
	lambda: GetEventSourceMapping
	lambda: GetEventSourceMapping
	lambda: GetFunction
	lambda: GetFunction
	lambda: GetFunction
	lambda: GetFunctionCodeSigningConfig
	lambda: GetFunctionConcurrency
	lambda: GetFunctionConfiguration
	lambda: GetFunctionConfiguration

Awalan layanan	Tindakan
	lambda: GetFunctionConfiguration lambda: GetFunctionEventInvokeConfig lambda: GetFunctionRecursionConfig lambda: GetFunctionUrlConfig lambda: GetLayerVersion lambda: GetLayerVersion lambda: GetLayerVersion lambda: GetLayerVersion lambda: GetLayerVersionPolicy lambda: GetLayerVersionPolicy lambda: GetPolicy lambda: GetPolicy lambda: GetPolicy lambda: GetProvisionedConcurrencyConfig lambda: GetRuntimeManagementConfig lambda: ListAliases lambda: ListAliases lambda: ListCodeSigningConfigs lambda: ListEventSourceMappings lambda: ListEventSourceMappings lambda: ListFunctionEventInvokeConfigs

Awalan layanan	Tindakan
	lambda: ListFunctions lambda: ListFunctions lambda: ListFunctionsByCodeSigningConfig lambda: ListFunctionUrlConfigs lambda: ListLayers lambda: ListLayers lambda: ListLayerVersions lambda: ListLayerVersions lambda: ListProvisionedConcurrencyConfigs lambda: ListVersionsByFunction lambda: ListVersionsByFunction lambda: PublishLayerVersion lambda: PublishLayerVersion lambda: PublishVersion lambda: PublishVersion lambda: PutFunctionCodeSigningConfig lambda: PutFunctionConcurrency lambda: PutFunctionConcurrency lambda: PutFunctionEventInvokeConfig lambda: PutFunctionRecursionConfig lambda: PutProvisionedConcurrencyConfig

Awalan layanan	Tindakan
	lambda: PutRuntimeManagementConfig lambda: RemoveLayerVersionPermission lambda: RemoveLayerVersionPermission lambda: RemovePermission lambda: RemovePermission lambda: RemovePermission lambda: UpdateAlias lambda: UpdateAlias lambda: UpdateCodeSigningConfig lambda: UpdateEventSourceMapping lambda: UpdateEventSourceMapping lambda: UpdateFunctionCode lambda: UpdateFunctionCode lambda: UpdateFunctionCode lambda: UpdateFunctionConfiguration lambda: UpdateFunctionConfiguration lambda: UpdateFunctionConfiguration lambda: UpdateFunctionEventInvokeConfig lambda: UpdateFunctionUrlConfig

Awalan layanan	Tindakan
lex	lex: BatchCreateCustomVocabularyItem
	lex: BatchDeleteCustomVocabularyItem
	lex: BatchUpdateCustomVocabularyItem
	lex: BuildBotLocale
	lex: CreateBotAlias
	lex: CreateBotReplica
	lex: CreateBotVersion
	lex: CreateExport
	lex: CreateIntentVersion
	lex: CreateResourcePolicy
	lex: CreateSlotTypeVersion
	lex: CreateTestSetDiscrepancyReport
	lex: CreateUploadUrl
	lex: DeleteBot
	lex: DeleteBotChannelAssociation
	lex: DeleteBotReplica
	lex: DeleteExport
	lex: DeleteImport
	lex: DeleteIntentVersion
	lex: DeleteResourcePolicy
	lex: DeleteSlotTypeVersion

Awalan layanan	Tindakan
	lex: DeleteTestSet
	lex: DeleteUtterances
	lex: DescribeBotAlias
	lex: DescribeBotRecommendation
	lex: DescribeBotReplica
	lex: DescribeBotResourceGeneration
	lex: DescribeBotVersion
	lex: DescribeCustomVocabularyMetadata
	lex: DescribeExport
	lex: DescribeImport
	lex: DescribeResourcePolicy
	lex: DescribeTestExecution
	lex: DescribeTestSet
	lex: DescribeTestSetDiscrepancyReport
	lex: DescribeTestSetGeneration
	lex: GenerateBotElement
	lex: GetBot
	lex: GetBotAlias
	lex: GetBotAliases
	lex: GetBotChannelAssociation
	lex: GetBotChannelAssociations

Awalan layanan	Tindakan
	lex: GetBots
	lex: GetBotVersions
	lex: GetBuiltinIntent
	lex: GetBuiltinIntents
	lex: GetBuiltinSlotTypes
	lex: GetExport
	lex: GetImport
	lex: GetIntent
	lex: GetIntents
	lex: GetIntentVersions
	lex: GetMigration
	lex: GetMigrations
	lex: GetSlotType
	lex: GetSlotTypes
	lex: GetSlotTypeVersions
	lex: GetTestExecutionArtifactsUrl
	lex: GetUtterancesView
	lex: ListBotAliases
	lex: ListBotAliasReplicas
	lex: ListBotRecommendations
	lex: ListBotReplicas

Awalan layanan	Tindakan
	lex: ListBotResourceGenerations
	lex: ListBots
	lex: ListBotVersionReplicas
	lex: ListBotVersions
	lex: ListBuiltInIntents
	lex: ListBuiltInSlotTypes
	lex: ListCustomVocabularyItems
	lex: ListExports
	lex: ListImports
	lex: ListIntentMetrics
	lex: ListIntentPaths
	lex: ListRecommendedIntents
	lex: ListSessionAnalyticsData
	lex: ListSessionMetrics
	lex: ListTestExecutionResultItems
	lex: ListTestExecutions
	lex: ListTestSets
	lex: PutBot
	lex: PutBotAlias
	lex: PutIntent
	lex: PutSlotType

Awalan layanan	Tindakan
	lex: SearchAssociatedTranscripts lex: StartBotRecommendation lex: StartImport lex: StartMigration lex: StartTestExecution lex: StartTestSetGeneration lex: StopBotRecommendation lex: UpdateBotAlias lex: UpdateBotRecommendation lex: UpdateExport lex: UpdateResourcePolicy
license-manager-linux-subscriptions	license-manager-linux-subscriptions:DeregisterSubscriptionProvider license-manager-linux-subscriptions:GetRegisteredSubscriptionProvider license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions:ListRegisteredSubscriptionProviders license-manager-linux-subscriptions:RegisterSubscriptionProvider license-manager-linux-subscriptions:UpdateServiceSettings

Awalan layanan	Tindakan
lightsail	lightsail: AllocateStaticIp
	lightsail: AttachCertificateToDistribution
	lightsail: AttachDisk
	lightsail: AttachInstancesToLoadBalancer
	lightsail: AttachLoadBalancerTlsCertificate
	lightsail: AttachStaticIp
	lightsail: CloseInstancePublicPorts
	lightsail: CopySnapshot
	lightsail: CreateBucket
	lightsail: CreateBucketAccessKey
	lightsail: CreateCertificate
	lightsail: CreateCloudFormationStack
	lightsail: CreateContactMethod
	lightsail: CreateContainerService
	lightsail: CreateContainerServiceDeployment
	lightsail: CreateContainerServiceRegistryLogin
	lightsail: CreateDisk
	lightsail: CreateDiskFromSnapshot
	lightsail: CreateDiskSnapshot
	lightsail: CreateDistribution
	lightsail: CreateDomain

Awalan layanan	Tindakan
	LightSail: C reateGUISession AccessDetails
	lightsail: CreateInstances
	lightsail: CreateInstancesFromSnapshot
	lightsail: CreateInstanceSnapshot
	lightsail: CreateKeyPair
	lightsail: CreateLoadBalancer
	lightsail: CreateLoadBalancerTlsCertificate
	lightsail: CreateRelationalDatabase
	lightsail: CreateRelationalDatabaseFromSnapshot
	lightsail: CreateRelationalDatabaseSnapshot
	lightsail: DeleteAlarm
	lightsail: DeleteAutoSnapshot
	lightsail: DeleteBucket
	lightsail: DeleteBucketAccessKey
	lightsail: DeleteCertificate
	lightsail: DeleteContactMethod
	lightsail: DeleteContainerImage
	lightsail: DeleteContainerService
	lightsail: DeleteDisk
	lightsail: DeleteDiskSnapshot
	lightsail: DeleteDistribution

Awalan layanan	Tindakan
	lightsail: DeleteDomain
	lightsail: DeleteDomainEntry
	lightsail: DeleteInstance
	lightsail: DeleteInstanceSnapshot
	lightsail: DeleteKeyPair
	lightsail: DeleteKnownHostKeys
	lightsail: DeleteLoadBalancer
	lightsail: DeleteLoadBalancerTlsCertificate
	lightsail: DeleteRelationalDatabase
	lightsail: DeleteRelationalDatabaseSnapshot
	lightsail: DetachCertificateFromDistribution
	lightsail: DetachDisk
	lightsail: DetachInstancesFromLoadBalancer
	lightsail: DetachStaticIp
	lightsail: DisableAddOn
	lightsail: DownloadDefaultKeyPair
	lightsail: EnableAddOn
	lightsail: ExportSnapshot
	lightsail: GetActiveNames
	lightsail: GetAlarms
	lightsail: GetAutoSnapshots

Awalan layanan	Tindakan
	lightsail: GetBlueprints
	lightsail: GetBucketAccessKeys
	lightsail: GetBucketBundles
	lightsail: GetBucketMetricData
	lightsail: GetBuckets
	lightsail: GetBundles
	lightsail: GetCertificates
	lightsail: GetCloudFormationStackRecords
	lightsail: GetContactMethods
	lightsail: GetContainer APIMetadata
	lightsail: GetContainerImages
	lightsail: GetContainerLog
	lightsail: GetContainerServiceDeployments
	lightsail: GetContainerServiceMetricData
	lightsail: GetContainerServicePowers
	lightsail: GetContainerServices
	lightsail: GetCostEstimate
	lightsail: GetDisk
	lightsail: GetDisks
	lightsail: GetDiskSnapshot
	lightsail: GetDiskSnapshots

Awalan layanan	Tindakan
	lightsail: GetDistributionBundles
	lightsail: GetDistributionLatestCacheReset
	lightsail: GetDistributionMetricData
	lightsail: GetDistributions
	lightsail: GetDomain
	lightsail: GetExportSnapshotRecords
	lightsail: GetInstance
	lightsail: GetInstanceMetricData
	lightsail: GetInstancePortStates
	lightsail: GetInstances
	lightsail: GetInstanceSnapshot
	lightsail: GetInstanceSnapshots
	lightsail: GetInstanceState
	lightsail: GetKeyPair
	lightsail: GetKeyPairs
	lightsail: GetLoadBalancer
	lightsail: GetLoadBalancerMetricData
	lightsail: GetLoadBalancers
	lightsail: GetLoadBalancerTlsCertificates
	lightsail: GetLoadBalancerTlsPolicies
	lightsail: GetOperation

Awalan layanan	Tindakan
	lightsail: GetOperations
	lightsail: GetOperationsForResource
	lightsail: GetRegions
	lightsail: GetRelationalDatabase
	lightsail: GetRelationalDatabaseBlueprints
	lightsail: GetRelationalDatabaseBundles
	lightsail: GetRelationalDatabaseEvents
	lightsail: GetRelationalDatabaseLogEvents
	lightsail: GetRelationalDatabaseLogStreams
	lightsail: GetRelationalDatabaseMasterUserPassword
	lightsail: GetRelationalDatabaseMetricData
	lightsail: GetRelationalDatabaseParameters
	lightsail: GetRelationalDatabases
	lightsail: GetRelationalDatabaseSnapshot
	lightsail: GetRelationalDatabaseSnapshots
	lightsail: GetSetupHistory
	lightsail: GetStaticIp
	lightsail: GetStaticIps
	lightsail: ImportKeyPair
	lightsail: IsVpcPeered
	lightsail: OpenInstancePublicPorts

Awalan layanan	Tindakan
	lightsail: PeerVpc
	lightsail: PutAlarm
	lightsail: PutInstancePublicPorts
	lightsail: RebootInstance
	lightsail: RebootRelationalDatabase
	lightsail: RegisterContainerImage
	lightsail: ReleaseStaticIp
	lightsail: ResetDistributionCache
	lightsail: SendContactMethodVerification
	lightsail: SetIpAddressType
	lightsail: SetResourceAccessForBucket
	lightsail: SetupInstanceHttps
	LightSail: S tartGUISession
	lightsail: StartInstance
	lightsail: StartRelationalDatabase
	LightSail: S topGUISession
	lightsail: StopInstance
	lightsail: StopRelationalDatabase
	lightsail: TestAlarm
	lightsail: UnpeerVpc
	lightsail: UpdateBucket

Awalan layanan	Tindakan
	lightsail: UpdateBucketBundle
	lightsail: UpdateContainerService
	lightsail: UpdateDistribution
	lightsail: UpdateDistributionBundle
	lightsail: UpdateDomainEntry
	lightsail: UpdateInstanceMetadataOptions
	lightsail: UpdateLoadBalancerAttribute
	lightsail: UpdateRelationalDatabase
	lightsail: UpdateRelationalDatabaseParameters

Awalan layanan	Tindakan
log	log: AssociateKmsKey log: CancelExportTask log: CreateDelivery log: CreateExportTask log: CreateLogAnomalyDetector log: CreateLogGroup log: CreateLogStream log: DeleteDataProtectionPolicy log: DeleteDelivery log: DeleteDeliveryDestination log: DeleteDeliveryDestinationPolicy log: DeleteDeliverySource log: DeleteDestination log: DeleteLogAnomalyDetector log: DeleteLogGroup log: DeleteLogStream log: DeleteMetricFilter log: DeleteQueryDefinition log: DeleteResourcePolicy log: DeleteRetentionPolicy log: DeleteSubscriptionFilter

Awalan layanan	Tindakan
	log: DescribeAccountPolicies log: DescribeConfigurationTemplates log: DescribeDeliveries log: DescribeDeliveryDestinations log: DescribeDeliverySources log: DescribeDestinations log: DescribeExportTasks log: DescribeLogGroups log: DescribeLogStreams log: DescribeMetricFilters log: DescribeQueries log: DescribeQueryDefinitions log: DescribeResourcePolicies log: DescribeSubscriptionFilters log: DisassociateKmsKey log: GetDataProtectionPolicy log: GetDelivery log: GetDeliveryDestination log: GetDeliveryDestinationPolicy log: GetDeliverySource log: GetLogAnomalyDetector

Awalan layanan	Tindakan
	log: GetLogGroupFields log: GetLogRecord log: GetQueryResults log: ListAnomalies log: ListLogAnomalyDetectors log: PutDataProtectionPolicy log: PutDeliveryDestination log: PutDeliveryDestinationPolicy log: PutDeliverySource log: PutDestination log: PutDestinationPolicy log: PutMetricFilter log: PutQueryDefinition log: PutResourcePolicy log: PutRetentionPolicy log: PutSubscriptionFilter log: StartLiveTail log: StartQuery log: StopQuery log: TestMetricFilter log: UpdateAnomaly

Awalan layanan	Tindakan
	log: UpdateDeliveryConfiguration log: UpdateLogAnomalyDetector

Awalan layanan	Tindakan
peralatan pencarian	Lookoutequipment: CreateDataset
	Lookoutequipment: CreateInferenceScheduler
	Lookoutequipment: CreateLabel
	Lookoutequipment: CreateLabelGroup
	Lookoutequipment: CreateModel
	Lookoutequipment: DeleteDataset
	Lookoutequipment: DeleteInferenceScheduler
	Lookoutequipment: DeleteLabel
	Lookoutequipment: DeleteLabelGroup
	Lookoutequipment: DeleteModel
	Lookoutequipment: DeleteResourcePolicy
	Lookoutequipment: DeleteRetrainingScheduler
	Lookoutequipment: DescribeDataIngestionJob
	Lookoutequipment: DescribeDataset
	Lookoutequipment: DescribeInferenceScheduler
	LookouteQuipment:Describelabel
	Lookoutequipment: DescribeLabelGroup
	Lookoutequipment: DescribeModel
	Lookoutequipment: DescribeModelVersion
	Lookoutequipment: DescribeResourcePolicy
	Lookoutequipment: DescribeRetrainingScheduler

Awalan layanan	Tindakan
	Lookoutequipment: ImportDataset
	Lookoutequipment: ImportModelVersion
	Lookoutequipment: ListDataIngestionJobs
	Lookoutequipment: ListDatasets
	Lookoutequipment: ListInferenceEvents
	Lookoutequipment: ListInferenceExecutions
	Lookoutequipment: ListInferenceSchedulers
	Lookoutequipment: ListLabelGroups
	Lookoutequipment: ListLabels
	Lookoutequipment: ListModels
	Lookoutequipment: ListModelVersions
	Lookoutequipment: ListRetrainingSchedulers
	Lookoutequipment: ListSensorStatistics
	Lookoutequipment: PutResourcePolicy
	Lookoutequipment: StartDataIngestionJob
	Lookoutequipment: StartInferenceScheduler
	Lookoutequipment: StartRetrainingScheduler
	Lookoutequipment: StopInferenceScheduler
	Lookoutequipment: StopRetrainingScheduler
	Lookoutequipment: UpdateActiveModelVersion
	Lookoutequipment: UpdateInferenceScheduler

Awalan layanan	Tindakan
	Lookoutequipment: UpdateLabelGroup Lookoutequipment: UpdateModel Lookoutequipment: UpdateRetrainingScheduler

Awalan layanan	Tindakan
lookoutmetrics	lookoutmetrics: ActivateAnomalyDetector lookoutmetrics: BackTestAnomalyDetector lookoutmetrics: CreateAlert lookoutmetrics: CreateAnomalyDetector lookoutmetrics: CreateMetricSet lookoutmetrics: DeactivateAnomalyDetector lookoutmetrics: DeleteAlert lookoutmetrics: DeleteAnomalyDetector lookoutmetrics: DescribeAlert lookoutmetrics: DescribeAnomalyDetectionExecutions lookoutmetrics: DescribeAnomalyDetector lookoutmetrics: DescribeMetricSet lookoutmetrics: DetectMetricSetConfig lookoutmetrics: GetAnomalyGroup lookoutmetrics: GetDataQualityMetrics lookoutmetrics: GetFeedback lookoutmetrics: GetSampleData lookoutmetrics: ListAlerts lookoutmetrics: ListAnomalyDetectors lookoutmetrics: ListAnomalyGroupRelatedMetrics lookoutmetrics: ListAnomalyGroupSummaries

Awalan layanan	Tindakan
	lookoutmetrics: ListAnomalyGroupTimeSeries lookoutmetrics: ListMetricSets lookoutmetrics: PutFeedback lookoutmetrics: UpdateAlert lookoutmetrics: UpdateAnomalyDetector lookoutmetrics: UpdateMetricSet

Awalan layanan	Tindakan
lookoutvision	penglihatan penglihatan: CreateDataset
	penglihatan penglihatan: CreateModel
	penglihatan penglihatan: CreateProject
	penglihatan penglihatan: DeleteDataset
	penglihatan penglihatan: DeleteModel
	penglihatan penglihatan: DeleteProject
	penglihatan penglihatan: DescribeDataset
	penglihatan penglihatan: DescribeModel
	penglihatan penglihatan: DescribeModelPackagingJob
	penglihatan penglihatan: DescribeProject
	penglihatan penglihatan: DetectAnomalies
	penglihatan penglihatan: ListDatasetEntries
	penglihatan penglihatan: ListModelPackagingJobs
	penglihatan penglihatan: ListModels
	penglihatan penglihatan: ListProjects
	penglihatan penglihatan: StartModel
	penglihatan penglihatan: StartModelPackagingJob
	penglihatan penglihatan: StopModel
	penglihatan penglihatan: UpdateDatasetEntries

Awalan layanan	Tindakan
m2	m2: CancelBatchJobExecution m2: CreateApplication m2: CreateDataSetImportTask m2: CreateDeployment m2: CreateEnvironment m2: DeleteApplication m2: DeleteApplicationFromEnvironment m2: DeleteEnvironment m2: GetApplication m2: GetApplicationVersion m2: GetBatchJobExecution m2: GetDataSetDetails m2: GetDataSetImportTask m2: GetDeployment m2: GetEnvironment m2: GetSignedBluinsightsUrl m2: ListApplications m2: ListApplicationVersions m2: ListBatchJobDefinitions m2: ListBatchJobExecutions m2: ListBatchJobRestartPoints

Awalan layanan	Tindakan
	m2: ListDataSetImportHistory m2: ListDataSets m2: ListDeployments m2: ListEngineVersions m2: ListEnvironments m2: StartApplication m2: StartBatchJob m2: StopApplication m2: UpdateApplication m2: UpdateEnvironment

Awalan layanan	Tindakan
terkelolablockchain	terkelolablockchain: CreateAccessor
	terkelolablockchain: CreateMember
	terkelolablockchain: CreateNetwork
	terkelolablockchain: CreateNode
	terkelolablockchain: CreateProposal
	terkelolablockchain: DeleteAccessor
	terkelolablockchain: DeleteMember
	terkelolablockchain: DeleteNode
	terkelolablockchain: GetAccessor
	terkelolablockchain: GetMember
	terkelolablockchain: GetNetwork
	terkelolablockchain: GetNode
	terkelolablockchain: GetProposal
	terkelolablockchain: InvokeRpcPolygonMainnet
	terkelolablockchain: InvokeRpcPolygonMumbaiTestnet
	terkelolablockchain: ListAccessors
	terkelolablockchain: ListInvitations
	terkelolablockchain: ListMembers
	terkelolablockchain: ListNetworks
	terkelolablockchain: ListNodes
	terkelolablockchain: ListProposals

Awalan layanan	Tindakan
	terkelolablockchain: ListProposalVotes terkelolablockchain: RejectInvitation terkelolablockchain: UpdateMember terkelolablockchain: UpdateNode terkelolablockchain: VoteOnProposal

Awalan layanan	Tindakan
mediaconnect	mediaconnect: AddBridgeOutputs mediaconnect: AddBridgeSources mediaconnect: AddFlowMediaStreams mediaconnect: AddFlowOutputs mediaconnect: AddFlowSources mediaconnect: AddFlowVpcInterfaces mediaconnect: CreateBridge mediaconnect: CreateFlow mediaconnect: CreateGateway mediaconnect: DeleteBridge mediaconnect: DeleteFlow mediaconnect: DeleteGateway mediaconnect: DeregisterGatewayInstance mediaconnect: DescribeBridge mediaconnect: DescribeFlow mediaconnect: DescribeFlowSourceMetadata mediaconnect: DescribeGateway mediaconnect: DescribeGatewayInstance mediaconnect: DescribeOffering mediaconnect: DescribeReservation mediaconnect: GrantFlowEntitlements

Awalan layanan	Tindakan
	mediaconnect: ListBridges mediaconnect: ListEntitlements mediaconnect: ListFlows mediaconnect: ListGatewayInstances mediaconnect: ListGateways mediaconnect: ListOfferings mediaconnect: ListReservations mediaconnect: PurchaseOffering mediaconnect: RemoveBridgeOutput mediaconnect: RemoveBridgeSource mediaconnect: RemoveFlowMediaStream mediaconnect: RemoveFlowOutput mediaconnect: RemoveFlowSource mediaconnect: RemoveFlowVpcInterface mediaconnect: RevokeFlowEntitlement mediaconnect: StartFlow mediaconnect: StopFlow mediaconnect: UpdateBridge mediaconnect: UpdateBridgeOutput mediaconnect: UpdateBridgeSource mediaconnect: UpdateBridgeState

Awalan layanan	Tindakan
	mediaconnect: UpdateFlow mediaconnect: UpdateFlowEntitlement mediaconnect: UpdateFlowMediaStream mediaconnect: UpdateFlowOutput mediaconnect: UpdateFlowSource mediaconnect: UpdateGatewayInstance

Awalan layanan	Tindakan
mediaconvert	mediaconvert: AssociateCertificate
	mediaconvert: CancelJob
	mediaconvert: CreateJob
	mediaconvert: CreateJobTemplate
	mediaconvert: CreatePreset
	mediaconvert: CreateQueue
	mediaconvert: DeleteJobTemplate
	mediaconvert: DeletePolicy
	mediaconvert: DeletePreset
	mediaconvert: DeleteQueue
	mediaconvert: DescribeEndpoints
	mediaconvert: DisassociateCertificate
	mediaconvert: GetJob
	mediaconvert: GetJobTemplate
	mediaconvert: GetPolicy
	mediaconvert: GetPreset
	mediaconvert: GetQueue
	mediaconvert: ListJobs
	mediaconvert: ListJobTemplates
	mediaconvert: ListPresets
	mediaconvert: ListQueues

Awalan layanan	Tindakan
	mediaconvert: PutPolicy
	mediaconvert: SearchJobs
	mediaconvert: UpdateJobTemplate
	mediaconvert: UpdatePreset
	mediaconvert: UpdateQueue

Awalan layanan	Tindakan
medialive	medialive: AcceptInputDeviceTransfer medialive: BatchDelete medialive: BatchStart medialive: BatchStop medialive: BatchUpdateSchedule medialive: CancelInputDeviceTransfer medialive: ClaimDevice medialive: CreateChannel medialive: CreateCloudWatchAlarmTemplate medialive: CreateCloudWatchAlarmTemplateGroup medialive: CreateEventBridgeRuleTemplate medialive: CreateEventBridgeRuleTemplateGroup medialive: CreateInput medialive: CreateInputSecurityGroup medialive: CreateMultiplex medialive: CreateMultiplexProgram medialive: CreatePartnerInput medialive: CreateSignalMap medialive: DeleteChannel medialive: DeleteCloudWatchAlarmTemplate medialive: DeleteCloudWatchAlarmTemplateGroup

Awalan layanan	Tindakan
	medialive: DeleteEventBridgeRuleTemplate medialive: DeleteEventBridgeRuleTemplateGroup medialive: DeleteInput medialive: DeleteInputSecurityGroup medialive: DeleteMultiplex medialive: DeleteMultiplexProgram medialive: DeleteReservation medialive: DeleteSchedule medialive: DeleteSignalMap medialive: DescribeAccountConfiguration medialive: DescribeChannel medialive: DescribeInput medialive: DescribeInputDevice medialive: DescribeInputDeviceThumbnail medialive: DescribeInputSecurityGroup medialive: DescribeMultiplex medialive: DescribeMultiplexProgram medialive: DescribeOffering medialive: DescribeReservation medialive: DescribeSchedule medialive: DescribeThumbnails

Awalan layanan	Tindakan
	medialive: GetCloudWatchAlarmTemplate medialive: GetCloudWatchAlarmTemplateGroup medialive: GetEventBridgeRuleTemplate medialive: GetEventBridgeRuleTemplateGroup medialive: GetSignalMap medialive: ListChannels medialive: ListCloudWatchAlarmTemplateGroups medialive: ListCloudWatchAlarmTemplates medialive: ListEventBridgeRuleTemplateGroups medialive: ListEventBridgeRuleTemplates medialive: ListInputDevices medialive: ListInputDeviceTransfers medialive: ListInputs medialive: ListInputSecurityGroups medialive: ListMultiplexes medialive: ListMultiplexPrograms medialive: ListOfferings medialive: ListReservations medialive: ListSignalMaps medialive: PurchaseOffering medialive: RebootInputDevice

Awalan layanan	Tindakan
	medialive: RejectInputDeviceTransfer medialive: RestartChannelPipelines medialive: StartChannel medialive: StartDeleteMonitorDeployment medialive: StartInputDevice medialive: StartInputDeviceMaintenanceWindow medialive: StartMonitorDeployment medialive: StartMultiplex medialive: StartUpdateSignalMap medialive: StopChannel medialive: StopInputDevice medialive: StopMultiplex medialive: TransferInputDevice medialive: UpdateAccountConfiguration medialive: UpdateChannel medialive: UpdateChannelClass medialive: UpdateCloudWatchAlarmTemplate medialive: UpdateCloudWatchAlarmTemplateGroup medialive: UpdateEventBridgeRuleTemplate medialive: UpdateEventBridgeRuleTemplateGroup medialive: UpdateInput

Awalan layanan	Tindakan
	medialive: UpdateInputDevice medialive: UpdateInputSecurityGroup medialive: UpdateMultiplex medialive: UpdateMultiplexProgram medialive: UpdateReservation

Awalan layanan	Tindakan
mediastore	mediastore: CreateContainer
	mediastore: DeleteContainer
	mediastore: DeleteContainerPolicy
	mediastore: DeleteCorsPolicy
	mediastore: DeleteLifecyclePolicy
	mediastore: DeleteMetricPolicy
	mediastore: DescribeContainer
	mediastore: GetContainerPolicy
	mediastore: GetCorsPolicy
	mediastore: GetLifecyclePolicy
	mediastore: GetMetricPolicy
	mediastore: ListContainers
	mediastore: PutContainerPolicy
	mediastore: PutCorsPolicy
	mediastore: PutLifecyclePolicy
	mediastore: PutMetricPolicy
	mediastore: StartAccessLogging
	mediastore: StopAccessLogging

Awalan layanan	Tindakan
mediatailor	mediatailor: ConfigureLogsForPlaybackConfiguration mediatailor: CreateChannel mediatailor: CreateLiveSource mediatailor: CreatePrefetchSchedule mediatailor: CreateProgram mediatailor: CreateSourceLocation mediatailor: CreateVodSource mediatailor: DeleteChannel mediatailor: DeleteChannelPolicy mediatailor: DeleteLiveSource mediatailor: DeletePlaybackConfiguration mediatailor: DeletePrefetchSchedule mediatailor: DeleteProgram mediatailor: DeleteSourceLocation mediatailor: DeleteVodSource mediatailor: DescribeChannel mediatailor: DescribeLiveSource mediatailor: DescribeProgram mediatailor: DescribeSourceLocation mediatailor: DescribeVodSource mediatailor: GetChannelPolicy

Awalan layanan	Tindakan
	mediatailor: GetChannelSchedule mediatailor: GetPlaybackConfiguration mediatailor: GetPrefetchSchedule mediatailor: ListAlerts mediatailor: ListChannels mediatailor: ListLiveSources mediatailor: ListPlaybackConfigurations mediatailor: ListPrefetchSchedules mediatailor: ListSourceLocations mediatailor: ListVodSources mediatailor: PutChannelPolicy mediatailor: PutPlaybackConfiguration mediatailor: StartChannel mediatailor: StopChannel mediatailor: UpdateChannel mediatailor: UpdateLiveSource mediatailor: UpdateProgram mediatailor: UpdateSourceLocation mediatailor: UpdateVodSource

Awalan layanan	Tindakan
memorydb	memorydb: BatchUpdateCluster
	memorydb: CopySnapshot
	memorydb: CreateAcl
	memorydb: CreateCluster
	memorydb: CreateParameterGroup
	memorydb: CreateSnapshot
	memorydb: CreateSubnetGroup
	memorydb: CreateUser
	memorydb: DeleteAcl
	memorydb: DeleteCluster
	memorydb: DeleteParameterGroup
	memorydb: DeleteSnapshot
	memorydb: DeleteSubnetGroup
	memorydb: DeleteUser
	memorydb: DescribeAcls
	memorydb: DescribeClusters
	memorydb: DescribeEngineVersions
	memorydb: DescribeEvents
	memorydb: DescribeParameterGroups
	memorydb: DescribeParameters
	memorydb: DescribeReservedNodes

Awalan layanan	Tindakan
	memorydb: DescribeReservedNodesOfferings
	memorydb: DescribeServiceUpdates
	memorydb: DescribeSnapshots
	memorydb: DescribeSubnetGroups
	memorydb: DescribeUsers
	memorydb: FailoverShard
	memorydb: ListAllowedNodeTypeUpdates
	memorydb: PurchaseReservedNodesOffering
	memorydb: ResetParameterGroup
	memorydb: UpdateAcl
	memorydb: UpdateCluster
	memorydb: UpdateParameterGroup
	memorydb: UpdateSubnetGroup
	memorydb: UpdateUser

Awalan layanan	Tindakan
mgh	mgh: AssociateCreatedArtifact mgh: AssociateDiscoveredResource mgh: CreateHomeRegionControl mgh: CreateProgressUpdateStream mgh: DeleteHomeRegionControl mgh: DeleteProgressUpdateStream mgh: DescribeApplicationState mgh: DescribeHomeRegionControls mgh: DescribeMigrationTask mgh: DisassociateCreatedArtifact mgh: DisassociateDiscoveredResource mgh: GetHomeRegion mgh: ImportMigrationTask mgh: ListApplicationStates mgh: ListCreatedArtifacts mgh: ListDiscoveredResources mgh: ListMigrationTasks mgh: ListProgressUpdateStreams mgh: NotifyApplicationState mgh: NotifyMigrationTaskState mgh: PutResourceAttributes

Awalan layanan	Tindakan
mgn	mgn: ArchiveApplication
	mgn: ArchiveWave
	mgn: AssociateApplications
	mgn: AssociateSourceServers
	mgn: ChangeServerLifeCycleState
	mgn: CreateApplication
	mgn: CreateConnector
	mgn: CreateLaunchConfigurationTemplate
	mgn: CreateReplicationConfigurationTemplate
	mgn: CreateWave
	mgn: DeleteApplication
	mgn: DeleteConnector
	mgn: DeleteJob
	mgn: DeleteLaunchConfigurationTemplate
	mgn: DeleteReplicationConfigurationTemplate
	mgn: DeleteSourceServer
	mgn: DeleteVcenterClient
	mgn: DeleteWave
	mgn: DescribeJobLogItems
	mgn: DescribeJobs
	mgn: DescribeLaunchConfigurationTemplates

Awalan layanan	Tindakan
	mgn: DescribeReplicationConfigurationTemplates
	mgn: DescribeVcenterClients
	mgn: DisassociateApplications
	mgn: DisassociateSourceServers
	mgn: DisconnectFromService
	mgn: FinalizeCutover
	mgn: GetReplicationConfiguration
	mgn: InitializeService
	mgn: ListConnectors
	mgn: ListExportErrors
	mgn: ListExports
	mgn: ListImportErrors
	mgn: ListImports
	mgn: ListManagedAccounts
	mgn: ListSourceServerActions
	mgn: ListTemplateActions
	mgn: MarkAsArchived
	mgn: PauseReplication
	mgn: PutSourceServerAction
	mgn: PutTemplateAction
	mgn: RemoveSourceServerAction

Awalan layanan	Tindakan
	mgn: RemoveTemplateAction
	mgn: ResumeReplication
	mgn: RetryDataReplication
	mgn: StartCutover
	mgn: StartExport
	mgn: StartImport
	mgn: StartReplication
	mgn: StartTest
	mgn: StopReplication
	mgn: TerminateTargetInstances
	mgn: UnarchiveApplication
	mgn: UnarchiveWave
	mgn: UpdateApplication
	mgn: UpdateConnector
	mgn: UpdateLaunchConfigurationTemplate
	mgn: UpdateReplicationConfiguration
	mgn: UpdateReplicationConfigurationTemplate
	mgn: UpdateSourceServer
	mgn: UpdateSourceServerReplicationType
	mgn: UpdateWave

Awalan layanan	Tindakan
Migrationhub-strategi	strategi hub migrasi: GetAntiPattern strategi hub migrasi: GetApplicationComponentDetails strategi hub migrasi: GetApplicationComponentStrategies strategi hub migrasi: GetAssessment strategi hub migrasi: GetImportFileTask strategi hub migrasi: GetLatestAssessmentId strategi hub migrasi: GetMessage strategi hub migrasi: GetPortfolioPreferences strategi hub migrasi: GetPortfolioSummary strategi hub migrasi: GetRecommendationReportDetails strategi hub migrasi: GetServerDetails strategi hub migrasi: GetServerStrategies strategi hub migrasi: ListAnalyzableServers strategi hub migrasi: ListAntiPatterns strategi hub migrasi: ListApplicationComponents strategi hub migrasi: ListCollectors strategi hub migrasi: ListImportFileTask strategi hub migrasi: ListJarArtifacts strategi hub migrasi: ListServers strategi hub migrasi: PutLogData strategi hub migrasi: PutMetricData

Awalan layanan	Tindakan
	strategi hub migrasi: PutPortfolioPreferences strategi hub migrasi: RegisterCollector strategi hub migrasi: SendMessage strategi hub migrasi: StartAssessment strategi hub migrasi: StartImportFileTask strategi hub migrasi: StartRecommendationReportGeneration strategi hub migrasi: StopAssessment strategi hub migrasi: UpdateApplicationComponentConfig strategi hub migrasi: UpdateCollectorConfiguration strategi hub migrasi: UpdateServerConfig

Awalan layanan	Tindakan
penargetan seluler	penargetan seluler: CreateApp
	penargetan seluler: CreateCampaign
	penargetan seluler: CreateEmailTemplate
	penargetan seluler: CreateExportJob
	penargetan seluler: CreateImportJob
	penargetan seluler: CreateInAppTemplate
	penargetan seluler: CreateJourney
	penargetan seluler: CreatePushTemplate
	penargetan seluler: CreateRecommenderConfiguration
	penargetan seluler: CreateSegment
	penargetan seluler: CreateSmsTemplate
	penargetan seluler: CreateVoiceTemplate
	penargetan seluler: DeleteAdmChannel
	penargetan seluler: DeleteApnsChannel
	penargetan seluler: DeleteApnsSandboxChannel
	penargetan seluler: DeleteApnsVoipChannel
	penargetan seluler: DeleteApnsVoipSandboxChannel
	penargetan seluler: DeleteApp
	penargetan seluler: DeleteBaiduChannel
	penargetan seluler: DeleteCampaign
	penargetan seluler: DeleteEmailChannel

Awalan layanan	Tindakan
	penargetan seluler: DeleteEmailTemplate
	penargetan seluler: DeleteEndpoint
	penargetan seluler: DeleteEventStream
	penargetan seluler: DeleteGcmChannel
	penargetan seluler: DeleteInAppTemplate
	penargetan seluler: DeleteJourney
	penargetan seluler: DeletePushTemplate
	penargetan seluler: DeleteRecommenderConfiguration
	penargetan seluler: DeleteSegment
	penargetan seluler: DeleteSmsChannel
	penargetan seluler: DeleteSmsTemplate
	penargetan seluler: DeleteUserEndpoints
	penargetan seluler: DeleteVoiceChannel
	penargetan seluler: DeleteVoiceTemplate
	penargetan seluler: GetAdmChannel
	penargetan seluler: GetApnsChannel
	penargetan seluler: GetApnsSandboxChannel
	penargetan seluler: GetApnsVoipChannel
	penargetan seluler: GetApnsVoipSandboxChannel
	penargetan seluler: GetApp
	penargetan seluler: GetApplicationDateRangeKpi

Awalan layanan	Tindakan
	penargetan seluler: GetApplicationSettings
	penargetan seluler: GetApps
	penargetan seluler: GetBaiduChannel
	penargetan seluler: GetCampaign
	penargetan seluler: GetCampaignActivities
	penargetan seluler: GetCampaignDateRangeKpi
	penargetan seluler: GetCampaigns
	penargetan seluler: GetCampaignVersion
	penargetan seluler: GetCampaignVersions
	penargetan seluler: GetChannels
	penargetan seluler: GetEmailChannel
	penargetan seluler: GetEmailTemplate
	penargetan seluler: GetEndpoint
	penargetan seluler: GetEventStream
	penargetan seluler: GetExportJob
	penargetan seluler: GetExportJobs
	penargetan seluler: GetGcmChannel
	penargetan seluler: GetImportJob
	penargetan seluler: GetImportJobs
	penargetan seluler: GetInAppMessages
	penargetan seluler: GetInAppTemplate

Awalan layanan	Tindakan
	penargetan seluler: GetJourney
	penargetan seluler: GetJourneyDateRangeKpi
	penargetan seluler: GetJourneyExecutionActivityMetrics
	penargetan seluler: GetJourneyExecutionMetrics
	penargetan seluler: GetJourneyRunExecutionActivityMetrics
	penargetan seluler: GetJourneyRunExecutionMetrics
	penargetan seluler: GetJourneyRuns
	penargetan seluler: GetPushTemplate
	penargetan seluler: GetRecommenderConfiguration
	penargetan seluler: GetRecommenderConfigurations
	penargetan seluler: GetSegment
	penargetan seluler: GetSegmentExportJobs
	penargetan seluler: GetSegmentImportJobs
	penargetan seluler: GetSegments
	penargetan seluler: GetSegmentVersion
	penargetan seluler: GetSegmentVersions
	penargetan seluler: GetSmsChannel
	penargetan seluler: GetSmsTemplate
	penargetan seluler: GetUserEndpoints
	penargetan seluler: GetVoiceChannel
	penargetan seluler: GetVoiceTemplate

Awalan layanan	Tindakan
	penargetan seluler: ListJourneys
	penargetan seluler: ListTemplates
	penargetan seluler: ListTemplateVersions
	penargetan seluler: PhoneNumberValidate
	penargetan seluler: PutEventStream
	penargetan seluler: RemoveAttributes
	penargetan seluler: UpdateAdmChannel
	penargetan seluler: UpdateApnsChannel
	penargetan seluler: UpdateApnsSandboxChannel
	penargetan seluler: UpdateApnsVoipChannel
	penargetan seluler: UpdateApnsVoipSandboxChannel
	penargetan seluler: UpdateApplicationSettings
	penargetan seluler: UpdateBaiduChannel
	penargetan seluler: UpdateCampaign
	penargetan seluler: UpdateEmailChannel
	penargetan seluler: UpdateEmailTemplate
	penargetan seluler: UpdateEndpoint
	penargetan seluler: UpdateEndpointsBatch
	penargetan seluler: UpdateGcmChannel
	penargetan seluler: UpdateInAppTemplate
	penargetan seluler: UpdateJourney

Awalan layanan	Tindakan
	penargetan seluler: UpdateJourneyState
	penargetan seluler: UpdatePushTemplate
	penargetan seluler: UpdateRecommenderConfiguration
	penargetan seluler: UpdateSegment
	penargetan seluler: UpdateSmsChannel
	penargetan seluler: UpdateSmsTemplate
	penargetan seluler: UpdateTemplateActiveVersion
	penargetan seluler: UpdateVoiceChannel
	penargetan seluler: UpdateVoiceTemplate
	Penargetan Mobile:V erifyOTPMessage

Awalan layanan	Tindakan
mq	mq: CreateBroker mq: CreateConfiguration mq: CreateUser mq: DeleteBroker mq: DeleteUser mq: DescribeBroker mq: DescribeBrokerEngineTypes mq: DescribeBrokerInstanceOptions mq: DescribeConfiguration mq: DescribeConfigurationRevision mq: DescribeUser mq: ListBrokers mq: ListConfigurationRevisions mq: ListConfigurations mq: ListUsers MQ: Mempromosikan mq: RebootBroker mq: UpdateBroker mq: UpdateConfiguration mq: UpdateUser

Awalan layanan	Tindakan
manajer jaringan	manajer jaringan: AcceptAttachment
	manajer jaringan: AssociateConnectPeer
	manajer jaringan: AssociateCustomerGateway
	manajer jaringan: AssociateLink
	manajer jaringan: AssociateTransitGatewayConnectPeer
	manajer jaringan: CreateConnectAttachment
	manajer jaringan: CreateConnection
	manajer jaringan: CreateConnectPeer
	manajer jaringan: CreateCoreNetwork
	manajer jaringan: CreateDevice
	manajer jaringan: CreateGlobalNetwork
	manajer jaringan: CreateLink
	manajer jaringan: CreateSite
	manajer jaringan: CreateSiteToSiteVpnAttachment
	manajer jaringan: CreateTransitGatewayPeering
	manajer jaringan: CreateTransitGatewayRouteTableAttachment
	manajer jaringan: CreateVpcAttachment
	manajer jaringan: DeleteAttachment
	manajer jaringan: DeleteConnection
	manajer jaringan: DeleteConnectPeer
	manajer jaringan: DeleteCoreNetwork

Awalan layanan	Tindakan
	manajer jaringan: DeleteCoreNetworkPolicyVersion
	manajer jaringan: DeleteDevice
	manajer jaringan: DeleteGlobalNetwork
	manajer jaringan: DeleteLink
	manajer jaringan: DeletePeering
	manajer jaringan: DeleteResourcePolicy
	manajer jaringan: DeleteSite
	manajer jaringan: DeregisterTransitGateway
	manajer jaringan: DescribeGlobalNetworks
	manajer jaringan: DisassociateConnectPeer
	manajer jaringan: DisassociateCustomerGateway
	manajer jaringan: DisassociateLink
	manajer jaringan: DisassociateTransitGatewayConnectPeer
	manajer jaringan: ExecuteCoreNetworkChangeSet
	manajer jaringan: GetConnectAttachment
	manajer jaringan: GetConnections
	manajer jaringan: GetConnectPeer
	manajer jaringan: GetConnectPeerAssociations
	manajer jaringan: GetCoreNetwork
	manajer jaringan: GetCoreNetworkChangeEvents
	manajer jaringan: GetCoreNetworkChangeSet

Awalan layanan	Tindakan
	manajer jaringan: GetCoreNetworkPolicy
	manajer jaringan: GetCustomerGatewayAssociations
	manajer jaringan: GetDevices
	manajer jaringan: GetLinkAssociations
	manajer jaringan: GetLinks
	manajer jaringan: GetNetworkResourceCounts
	manajer jaringan: GetNetworkResourceRelationships
	manajer jaringan: GetNetworkResources
	manajer jaringan: GetNetworkRoutes
	manajer jaringan: GetNetworkTelemetry
	manajer jaringan: GetResourcePolicy
	manajer jaringan: GetRouteAnalysis
	manajer jaringan: GetSites
	manajer jaringan: GetSiteToSiteVpnAttachment
	manajer jaringan: GetTransitGatewayConnectPeerAssociations
	manajer jaringan: GetTransitGatewayPeering
	manajer jaringan: GetTransitGatewayRegistrations
	manajer jaringan: GetTransitGatewayRouteTableAttachment
	manajer jaringan: GetVpcAttachment
	manajer jaringan: ListAttachments
	manajer jaringan: ListConnectPeers

Awalan layanan	Tindakan
	manajer jaringan: ListCoreNetworkPolicyVersions
	manajer jaringan: ListCoreNetworks
	manajer jaringan: ListOrganizationServiceAccessStatus
	manajer jaringan: ListPeerings
	manajer jaringan: PutCoreNetworkPolicy
	manajer jaringan: PutResourcePolicy
	manajer jaringan: RegisterTransitGateway
	manajer jaringan: RejectAttachment
	manajer jaringan: RestoreCoreNetworkPolicyVersion
	manajer jaringan: StartOrganizationServiceAccessUpdate
	manajer jaringan: StartRouteAnalysis
	manajer jaringan: UpdateConnection
	manajer jaringan: UpdateCoreNetwork
	manajer jaringan: UpdateDevice
	manajer jaringan: UpdateGlobalNetwork
	manajer jaringan: UpdateLink
	manajer jaringan: UpdateNetworkResourceMetadata
	manajer jaringan: UpdateSite
	manajer jaringan: UpdateVpcAttachment

Awalan layanan	Tindakan
gesit	gesit: AcceptEulas gesit: CreateLaunchProfile gesit: CreateStreamingImage gesit: CreateStreamingSession gesit: CreateStreamingSessionStream gesit: CreateStudio gesit: CreateStudioComponent gesit: DeleteLaunchProfile gesit: DeleteLaunchProfileMember gesit: DeleteStreamingImage gesit: DeleteStreamingSession gesit: DeleteStudio gesit: DeleteStudioComponent gesit: DeleteStudioMember gesit: GetEula gesit: GetLaunchProfileDetails gesit: GetStreamingImage gesit: GetStreamingSession gesit: GetStreamingSessionBackup gesit: GetStreamingSessionStream gesit: GetStudio

Awalan layanan	Tindakan
	gesit: GetStudioComponent gesit: GetStudioMember gesit: ListEulas gesit: ListLaunchProfileMembers gesit: ListLaunchProfiles gesit: ListStreamingImages gesit: ListStreamingSessionBackups gesit: ListStreamingSessions gesit: ListStudioComponents gesit: ListStudioMembers gesit: ListStudios gesit: PutLaunchProfileMembers gesit: PutStudioMembers gesit: StartStreamingSession gesit: StartStudio SSOConfigurationRepair gesit: StopStreamingSession gesit: UpdateLaunchProfile gesit: UpdateLaunchProfileMember gesit: UpdateStreamingImage gesit: UpdateStudio gesit: UpdateStudioComponent

Awalan layanan	Tindakan
omics	omics: AbortMultipartReadSetUpload
	omics: BatchDeleteReadSet
	omics: CancelAnnotationImportJob
	omics: CancelRun
	omics: CancelVariantImportJob
	omics: CompleteMultipartReadSetUpload
	omics: CreateAnnotationStore
	omics: CreateMultipartReadSetUpload
	omics: CreateReferenceStore
	omics: CreateRunGroup
	omics: CreateSequenceStore
	omics: CreateVariantStore
	omics: CreateWorkflow
	omics: DeleteAnnotationStore
	omics: DeleteReference
	omics: DeleteReferenceStore
	omics: DeleteRun
	omics: DeleteRunGroup
	omics: DeleteSequenceStore
	omics: DeleteVariantStore
	omics: DeleteWorkflow

Awalan layanan	Tindakan
	omics: GetAnnotationImportJob
	omics: GetAnnotationStore
	omics: GetReadSet
	omics: GetReadSetActivationJob
	omics: GetReadSetExportJob
	omics: GetReadSetImportJob
	omics: GetReadSetMetadata
	omics: GetReference
	omics: GetReferenceImportJob
	omics: GetReferenceMetadata
	omics: GetReferenceStore
	omics: GetRun
	omics: GetRunGroup
	omics: GetRunTask
	omics: GetSequenceStore
	omics: GetVariantImportJob
	omics: GetVariantStore
	omics: GetWorkflow
	omics: ListAnnotationImportJobs
	omics: ListAnnotationStores
	omics: ListMultipartReadSetUploads

Awalan layanan	Tindakan
	omics: ListReadSetActivationJobs
	omics: ListReadSetExportJobs
	omics: ListReadSetImportJobs
	omics: ListReadSets
	omics: ListReadSetUploadParts
	omics: ListReferenceImportJobs
	omics: ListReferences
	omics: ListReferenceStores
	omics: ListRunGroups
	omics: ListRuns
	omics: ListRunTasks
	omics: ListSequenceStores
	omics: ListVariantImportJobs
	omics: ListVariantStores
	omics: ListWorkflows
	omics: StartAnnotationImportJob
	omics: StartReadSetActivationJob
	omics: StartReadSetExportJob
	omics: StartReadSetImportJob
	omics: StartReferenceImportJob
	omics: StartRun

Awalan layanan	Tindakan
	omics: StartVariantImportJob
	omics: UpdateAnnotationStore
	omics: UpdateRunGroup
	omics: UpdateVariantStore
	omics: UpdateWorkflow
	omics: UploadReadSetPart

Awalan layanan	Tindakan
opsworks	opsworks: AssignInstance opsworks: AssignVolume opsworks: AssociateElasticIp opsworks: AttachElasticLoadBalancer opsworks: CloneStack opsworks: CreateApp opsworks: CreateDeployment opsworks: CreateInstance opsworks: CreateLayer opsworks: CreateStack opsworks: CreateUserProfile opsworks: DeleteApp opsworks: DeleteInstance opsworks: DeleteLayer opsworks: DeleteStack opsworks: DeleteUserProfile opsworks: DeregisterEcsCluster opsworks: DeregisterElasticIp opsworks: DeregisterInstance opsworks: DeregisterRdsDbInstance opsworks: DeregisterVolume

Awalan layanan	Tindakan
	opsworks: DescribeAgentVersions opsworks: DescribeApps opsworks: DescribeCommands opsworks: DescribeDeployments opsworks: DescribeEcsClusters opsworks: DescribeElasticIps opsworks: DescribeElasticLoadBalancers opsworks: DescribeInstances opsworks: DescribeLayers opsworks: DescribeLoadBasedAutoScaling opsworks: DescribeMyUserProfile opsworks: DescribeOperatingSystems opsworks: DescribePermissions opsworks: DescribeRaidArrays opsworks: DescribeRdsDbInstances opsworks: DescribeServiceErrors opsworks: DescribeStackProvisioningParameters opsworks: DescribeStacks opsworks: DescribeStackSummary opsworks: DescribeTimeBasedAutoScaling opsworks: DescribeUserProfiles

Awalan layanan	Tindakan
	opsworks: DescribeVolumes opsworks: DetachElasticLoadBalancer opsworks: DisassociateElasticIp opsworks: GetHostnameSuggestion opsworks: GrantAccess opsworks: RebootInstance opsworks: RegisterEcsCluster opsworks: RegisterElasticIp opsworks: RegisterInstance opsworks: RegisterRdsDbInstance opsworks: RegisterVolume opsworks: SetLoadBasedAutoScaling opsworks: SetPermission opsworks: SetTimeBasedAutoScaling opsworks: StartInstance opsworks: StartStack opsworks: StopInstance opsworks: StopStack opsworks: UnassignInstance opsworks: UnassignVolume opsworks: UpdateApp

Awalan layanan	Tindakan
	opsworks: UpdateElasticIp opsworks: UpdateInstance opsworks: UpdateLayer opsworks: UpdateMyUserProfile opsworks: UpdateRdsDbInstance opsworks: UpdateStack opsworks: UpdateUserProfile opsworks: UpdateVolume

Awalan layanan	Tindakan
opsworks-cm	opsworks-cm: AssociateNode
	opsworks-cm: CreateBackup
	opsworks-cm: CreateServer
	opsworks-cm: DeleteBackup
	opsworks-cm: DeleteServer
	opsworks-cm: DescribeAccountAttributes
	opsworks-cm: DescribeBackups
	opsworks-cm: DescribeEvents
	opsworks-cm: DescribeNodeAssociationStatus
	opsworks-cm: DescribeServers
	opsworks-cm: DisassociateNode
	opsworks-cm: ExportServerEngineAttribute
	opsworks-cm: RestoreServer
	opsworks-cm: StartMaintenance
	opsworks-cm: UpdateServer
	opsworks-cm: UpdateServerEngineAttributes

Awalan layanan	Tindakan
organisasi	organisasi: AcceptHandshake organisasi: AttachPolicy organisasi: CancelHandshake organisasi: CloseAccount organisasi: CreateAccount organisasi: CreateGovCloudAccount organisasi: CreateOrganization organisasi: CreateOrganizationalUnit organisasi: CreatePolicy organisasi: DeclineHandshake organisasi: DeleteOrganization organisasi: DeleteOrganizationalUnit organisasi: DeletePolicy organisasi: DeleteResourcePolicy organisasi: DeregisterDelegatedAdministrator organisasi: DescribeAccount organisasi: DescribeCreateAccountStatus organisasi: DescribeEffectivePolicy organisasi: DescribeHandshake organisasi: DescribeOrganization organisasi: DescribeOrganizationalUnit

Awalan layanan	Tindakan
	organisasi: DescribePolicy organisasi: DescribeResourcePolicy organisasi: DetachPolicy OrganisasiisableAWSService: D Akses organisasi: DisablePolicyType organisasi: EnableAllFeatures Organisasi:e Access nableAWSService organisasi: EnablePolicyType organisasi: InviteAccountToOrganization organisasi: LeaveOrganization organisasi: ListAccounts organisasi: ListAccountsForParent Organisasi: L istAWSService AccessForOrganization organisasi: ListChildren organisasi: ListCreateAccountStatus organisasi: ListDelegatedAdministrators organisasi: ListDelegatedServicesForAccount organisasi: ListHandshakesForAccount organisasi: ListHandshakesForOrganization organisasi: ListOrganizationalUnitsForParent organisasi: ListParents

Awalan layanan	Tindakan
	organisasi: ListPolicies organisasi: ListPoliciesForTarget organisasi: ListRoots organisasi: ListTargetsForPolicy organisasi: MoveAccount organisasi: PutResourcePolicy organisasi: RegisterDelegatedAdministrator organisasi: RemoveAccountFromOrganization organisasi: UpdateOrganizationalUnit organisasi: UpdatePolicy

Awalan layanan	Tindakan
pos terdepan	pos terdepan: CancelCapacityTask
	pos terdepan: CancelOrder
	pos terdepan: CreateOrder
	pos terdepan: CreateOutpost
	pos terdepan: CreatePrivateConnectivityConfig
	pos terdepan: CreateSite
	pos terdepan: DeleteOutpost
	pos terdepan: DeleteSite
	pos terdepan: GetCapacityTask
	pos terdepan: GetCatalogItem
	pos terdepan: GetConnection
	pos terdepan: GetOrder
	pos terdepan: GetOutpost
	pos terdepan: GetOutpostInstanceTypes
	pos terdepan: GetOutpostSupportedInstanceTypes
	pos terdepan: GetPrivateConnectivityConfig
	pos terdepan: GetSite
	pos terdepan: GetSiteAddress
	pos terdepan: ListAssets
	pos terdepan: ListCapacityTasks
	pos terdepan: ListCatalogItems

Awalan layanan	Tindakan
	pos terdepan: ListOrders
	pos terdepan: ListOutposts
	pos terdepan: ListSites
	pos terdepan: StartCapacityTask
	pos terdepan: StartConnection
	pos terdepan: UpdateOutpost
	pos terdepan: UpdateSite
	pos terdepan: UpdateSiteAddress
	pos terdepan: UpdateSiteRackPhysicalProperties

Awalan layanan	Tindakan
panorama	panorama: CreateApplicationInstance panorama: CreateJobForDevices panorama: CreateNodeFromTemplateJob panorama: CreatePackage panorama: CreatePackageImportJob panorama: DeleteDevice panorama: DeletePackage panorama: DeregisterPackageVersion panorama: DescribeApplicationInstance panorama: DescribeApplicationInstanceDetails panorama: DescribeDevice panorama: DescribeDeviceJob panorama: DescribeNode panorama: DescribeNodeFromTemplateJob panorama: DescribePackage panorama: DescribePackageImportJob panorama: DescribePackageVersion panorama: ListApplicationInstanceDependencies panorama: ListApplicationInstanceNodeInstances panorama: ListApplicationInstances panorama: ListDevices

Awalan layanan	Tindakan
	panorama: ListDevicesJobs panorama: ListNodeFromTemplateJobs panorama: ListNodes panorama: ListPackageImportJobs panorama: ListPackages panorama: ProvisionDevice panorama: RegisterPackageVersion panorama: RemoveApplicationInstance panorama: SignalApplicationInstanceNodeInstances panorama: UpdateDeviceMetadata
pi	pi: CreatePerformanceAnalysisReport pi: DeletePerformanceAnalysisReport pi: DescribeDimensionKeys pi: GetDimensionKeyDetails pi: GetPerformanceAnalysisReport pi: GetResourceMetadata pi: GetResourceMetrics pi: ListAvailableResourceDimensions pi: ListAvailableResourceMetrics pi: ListPerformanceAnalysisReports

Awalan layanan	Tindakan
pipa	pipa: CreatePipe pipa: DeletePipe pipa: DescribePipe pipa: ListPipes pipa: StartPipe pipa: StopPipe pipa: UpdatePipe
polly	polly: DeleteLexicon polly: DescribeVoices polly: GetLexicon polly: GetSpeechSynthesisTask polly: ListLexicons polly: ListSpeechSynthesisTasks polly: PutLexicon polly: StartSpeechSynthesisTask polly: SynthesizeSpeech

Awalan layanan	Tindakan
profile	Profil: AddProfileKey Profil: CreateCalculatedAttributeDefinition Profil: CreateDomain Profil: CreateEventStream Profil: CreateProfile Profil: DeleteCalculatedAttributeDefinition Profil: DeleteDomain Profil: DeleteEventStream Profil: DeleteIntegration Profil: DeleteProfile Profil: DeleteProfileKey Profil: DeleteProfileObject Profil: DeleteProfileObjectType Profil: DeleteWorkflow Profil: DetectProfileObjectType Profil: GetAutoMergingPreview Profil: GetCalculatedAttributeDefinition Profil: GetCalculatedAttributeForProfile Profil: GetDomain Profil: GetEventStream Profil: GetIdentityResolutionJob

Awalan layanan	Tindakan
	Profil: GetIntegration Profil: GetMatches Profil: GetProfileObjectType Profil: GetProfileObjectTypeTemplate Profil: GetSimilarProfiles Profil: GetWorkflow Profil: GetWorkflowSteps Profil: ListAccountIntegrations Profil: ListCalculatedAttributeDefinitions Profil: ListCalculatedAttributesForProfile Profil: ListDomains Profil: ListEventStreams Profil: ListIdentityResolutionJobs Profil: ListIntegrations Profil: ListProfileObjects Profil: ListProfileObjectTypes Profil: ListProfileObjectTypeTemplates Profil: ListRuleBasedMatches Profil: ListWorkflows Profil: MergeProfiles Profil: PutIntegration

Awalan layanan	Tindakan
	Profil: PutProfileObject Profil: PutProfileObjectType Profil: SearchProfiles Profil: UpdateCalculatedAttributeDefinition Profil: UpdateDomain Profil: UpdateProfile

Awalan layanan	Tindakan
qldb	qldb: CancelJournalKinesisStream
	qldb: CreateLedger
	qldb: DeleteLedger
	qldb: DescribeJournalKinesisStream
	qldb: S3Ekspor DescribeJournal
	qldb: DescribeLedger
	qldb: S3 ExportJournalTo
	qldb: GetBlock
	qldb: GetDigest
	qldb: GetRevision
	qldb: ListJournalKinesisStreamsForLedger
	qldb: S3Ekspor ListJournal
	qldb: S3 ListJournal ExportsForLedger
	qldb: ListLedgers
	qldb: StreamJournalToKinesis
	qldb: UpdateLedger
	qldb: UpdateLedgerPermissionsMode

Awalan layanan	Tindakan
ram	ram: AcceptResourceShareInvitation ram: AssociateResourceShare ram: AssociateResourceSharePermission ram: CreatePermission ram: CreatePermissionVersion ram: CreateResourceShare ram: DeletePermission ram: DeletePermissionVersion ram: DeleteResourceShare ram: DisassociateResourceShare ram: DisassociateResourceSharePermission ram: EnableSharingWithAwsOrganization ram: GetPermission ram: GetResourcePolicies ram: GetResourceShareAssociations ram: GetResourceShareInvitations ram: GetResourceShares ram: ListPendingInvitationResources ram: ListPermissionAssociations ram: ListPermissions ram: ListPermissionVersions

Awalan layanan	Tindakan
	ram: ListPrincipals ram: ListReplacePermissionAssociationsWork ram: ListResources ram: ListResourceSharePermissions ram: ListResourceTypes ram: PromotePermissionCreatedFromPolicy ram: PromoteResourceShareCreatedFromPolicy ram: RejectResourceShareInvitation ram: ReplacePermissionAssociations ram: SetDefaultPermissionVersion ram: UpdateResourceShare
rbin	rbin: CreateRule rbin: DeleteRule rbin: GetRule rbin: ListRules rbin: LockRule rbin: UnlockRule rbin: UpdateRule

Awalan layanan	Tindakan
rds	rds: AddRoleTo DBCluster rds: AddRoleTo DBInstance rds: AddSourceIdentifierToSubscription rds: ApplyPendingMaintenanceAction RDS: A uthorizeDBSecurity GroupIngress RDS: B acktrackDBCluster rds: CancelExportTask RDS: C opyDBCluster ParameterGroup RDSopyDBCluster: C Snapshot RDSopyDBParameter: Grup C RDS: C opyDBSnapshot rds: CopyOptionGroup rds: CreateCustom DBEngineVersion RDS: C reateDBCluster ParameterGroup RDSreateDBParameter: Grup C RDS: C reateDBProxy RDSreateDBProxy: C Titik Akhir RDSreateDBSecurity: Grup C RDSreateDBSubnet: Grup C rds: CreateEventSubscription rds: CreateGlobalCluster

Awalan layanan	Tindakan
	rds: CreateOptionGroup rds: DeleteBlueGreenDeployment RDS: D eleteDBCluster AutomatedBackup RDS: D eleteDBCluster ParameterGroup RDSeleteDBCluster: D Snapshot RDS: D eleteDBInstance AutomatedBackup RDSeleteDBParameter: Grup D RDS: D eleteDBProxy RDSeleteDBProxy: D Titik Akhir RDSeleteDBSecurity: Grup D RDS: D eleteDBSnapshot RDSeleteDBSubnet: Grup D rds: DeleteEventSubscription rds: DeleteGlobalCluster rds: DeleteOptionGroup RDSeregisterDBProxy: D Target rds: DescribeAccountAttributes rds: DescribeBlueGreenDeployments rds: DescribeCertificates RDS: D escribeDBCluster AutomatedBackups RDSescribeDBCluster: D Backtrack

Awalan layanan	Tindakan
	RDSescribeDBCluster: D Titik Akhir RDS: D escribeDBCluster ParameterGroups RDSescribeDBCluster: D Parameter RDS: D escribeDBClusters RDS: D escribeDBCluster SnapshotAttributes RDSescribeDBCluster: D Snapshot RDSescribeDBEngine: Versi D RDS: D escribeDBInstance AutomatedBackups RDS: D escribeDBInstances RDSescribeDBLog: D File RDSescribeDBParameter: Grup D RDS: D escribeDBParameters RDS: D escribeDBProxies RDSescribeDBProxy: D Titik Akhir RDS: D escribeDBProxy TargetGroups RDSescribeDBProxy: D Target RDS: D escribeDBRecommendations RDSescribeDBSecurity: Grup D RDSescribeDBSnapshot: D Atribut RDS: D escribeDBSnapshots RDS: D escribeDBSnapshot TenantDatabases

Awalan layanan	Tindakan
	RDSescribeDBSubnet: Grup D rds: DescribeEngineDefaultClusterParameters rds: DescribeEngineDefaultParameters rds: DescribeEventCategories rds: DescribeEvents rds: DescribeEventSubscriptions rds: DescribeExportTasks rds: DescribeGlobalClusters rds: DescribeIntegrations rds: DescribeOptionGroupOptions rds: DescribeOptionGroups rds: DescribeOrderable DBInstanceOptions rds: DescribePendingMaintenanceActions rds: DescribeReserved DBInstances rds: DescribeReserved DBInstancesOfferings rds: DescribeSourceRegions rds: DescribeTenantDatabases rds: DescribeValid DBInstanceModifications rds: DownloadComplete DBLogFile RDS: D ownloadDBLog FilePortion RDS: F ailoverDBCluster

Awalan layanan	Tindakan
	rds: FailoverGlobalCluster rds: ModifyActivityStream rds: ModifyCertificates rds: ModifyCurrent DBClusterCapacity RDSodifyDBCluster: M Titik Akhir RDS: M odifyDBCluster ParameterGroup RDS: M odifyDBCluster SnapshotAttribute RDSodifyDBParameter: Grup M RDS: M odifyDBProxy RDSodifyDBProxy: M Titik Akhir RDS: M odifyDBProxy TargetGroup RDS: M odifyDBRecommendation RDS: M odifyDBSnapshot RDSodifyDBSnapshot: M Atribut RDSodifyDBSubnet: Grup M rds: ModifyEventSubscription rds: ModifyGlobalCluster rds: ModifyOptionGroup rds: ModifyTenantDatabase rds: PurchaseReserved DBInstancesOffering RDS: R ebootDBCluster

Awalan layanan	Tindakan
	RDSegisterDBProxy: R Target rds: RemoveFromGlobalCluster rds: RemoveRoleFrom DBCluster rds: RemoveRoleFrom DBInstance rds: RemoveSourceIdentifierFromSubscription RDS: R esetDBCluster ParameterGroup RDSesetDBParameter: Grup R RDS: R dariMs3 estoreDBCluster RDS: R estoreDBCluster FromSnapshot RDS: R estoreDBCluster ToPointInTime RDSestoreDBInstance: R F romDBSnapshot RDS: R dariMs3 estoreDBInstance RDS: R estoreDBInstance ToPointInTime RDS: R evokeDBSecurity GroupIngress rds: StartActivityStream RDS: S tartDBCluster RDS: S tartDBInstance RDS: S tartDBInstance AutomatedBackupsReplication rds: StartExportTask rds: StopActivityStream RDS: S topDBCluster

Awalan layanan	Tindakan
	RDS: S topDBInstance RDS: S topDBInstance AutomatedBackupsReplication rds: SwitchoverBlueGreenDeployment rds: SwitchoverGlobalCluster rds: SwitchoverReadReplica

Awalan layanan	Tindakan
redshift	pergeseran merah: AcceptReservedNodeExchange
	pergeseran merah: AddPartner
	pergeseran merah: AssociateDataShareConsumer
	pergeseran merah: AuthorizeClusterSecurityGroupIngress
	pergeseran merah: AuthorizeDataShare
	pergeseran merah: AuthorizeEndpointAccess
	pergeseran merah: AuthorizeSnapshotAccess
	pergeseran merah: BatchDeleteClusterSnapshots
	pergeseran merah: BatchModifyClusterSnapshots
	pergeseran merah: CancelResize
	pergeseran merah: CopyClusterSnapshot
	pergeseran merah: CreateAuthenticationProfile
	pergeseran merah: CreateCluster
	pergeseran merah: CreateClusterParameterGroup
	pergeseran merah: CreateClusterSecurityGroup
	pergeseran merah: CreateClusterSnapshot
	pergeseran merah: CreateClusterSubnetGroup
	pergeseran merah: CreateCustomDomainAssociation
	pergeseran merah: CreateEndpointAccess
	pergeseran merah: CreateEventSubscription
	pergeseran merah: CreateHsmClientCertificate

Awalan layanan	Tindakan
	pergeseran merah: CreateHsmConfiguration
	pergeseran merah: CreateRedshiftIdcApplication
	pergeseran merah: CreateScheduledAction
	pergeseran merah: CreateSnapshotCopyGrant
	pergeseran merah: CreateSnapshotSchedule
	pergeseran merah: CreateUsageLimit
	pergeseran merah: DeauthorizeDataShare
	pergeseran merah: DeleteAuthenticationProfile
	pergeseran merah: DeleteCluster
	pergeseran merah: DeleteClusterParameterGroup
	pergeseran merah: DeleteClusterSecurityGroup
	pergeseran merah: DeleteClusterSnapshot
	pergeseran merah: DeleteClusterSubnetGroup
	pergeseran merah: DeleteCustomDomainAssociation
	pergeseran merah: DeleteEndpointAccess
	pergeseran merah: DeleteEventSubscription
	pergeseran merah: DeleteHsmClientCertificate
	pergeseran merah: DeleteHsmConfiguration
	pergeseran merah: DeletePartner
	pergeseran merah: DeleteRedshiftIdcApplication
	pergeseran merah: DeleteResourcePolicy

Awalan layanan	Tindakan
	pergeseran merah: DeleteScheduledAction
	pergeseran merah: DeleteSnapshotCopyGrant
	pergeseran merah: DeleteSnapshotSchedule
	pergeseran merah: DeleteUsageLimit
	pergeseran merah: DescribeAccountAttributes
	pergeseran merah: DescribeAuthenticationProfiles
	pergeseran merah: DescribeClusterDbRevisions
	pergeseran merah: DescribeClusterParameterGroups
	pergeseran merah: DescribeClusterParameters
	pergeseran merah: DescribeClusters
	pergeseran merah: DescribeClusterSecurityGroups
	pergeseran merah: DescribeClusterSnapshots
	pergeseran merah: DescribeClusterSubnetGroups
	pergeseran merah: DescribeClusterTracks
	pergeseran merah: DescribeClusterVersions
	pergeseran merah: DescribeCustomDomainAssociations
	pergeseran merah: DescribeDataShares
	pergeseran merah: DescribeDataSharesForConsumer
	pergeseran merah: DescribeDataSharesForProducer
	pergeseran merah: DescribeDefaultClusterParameters
	pergeseran merah: DescribeEndpointAccess

Awalan layanan	Tindakan
	pergeseran merah: DescribeEndpointAuthorization
	pergeseran merah: DescribeEventCategories
	pergeseran merah: DescribeEvents
	pergeseran merah: DescribeEventSubscriptions
	pergeseran merah: DescribeHsmClientCertificates
	pergeseran merah: DescribeHsmConfigurations
	pergeseran merah: DescribeInboundIntegrations
	pergeseran merah: DescribeLoggingStatus
	pergeseran merah: DescribeNodeConfigurationOptions
	pergeseran merah: DescribeOrderableClusterOptions
	pergeseran merah: DescribePartners
	pergeseran merah: DescribeRedshiftIdcApplications
	pergeseran merah: DescribeReservedNodeExchangeStatus
	pergeseran merah: DescribeReservedNodeOfferings
	pergeseran merah: DescribeReservedNodes
	pergeseran merah: DescribeResize
	pergeseran merah: DescribeScheduledActions
	pergeseran merah: DescribeSnapshotCopyGrants
	pergeseran merah: DescribeSnapshotSchedules
	pergeseran merah: DescribeStorage
	pergeseran merah: DescribeTableRestoreStatus

Awalan layanan	Tindakan
	pergeseran merah: DescribeUsageLimits
	pergeseran merah: DisableLogging
	pergeseran merah: DisableSnapshotCopy
	pergeseran merah: DisassociateDataShareConsumer
	pergeseran merah: EnableLogging
	pergeseran merah: EnableSnapshotCopy
	pergeseran merah: FailoverPrimaryCompute
	pergeseran merah: GetClusterCredentials
	pergeseran merah: GetClusterCredentialsWith IAM
	pergeseran merah: GetReservedNodeExchangeConfiguration Options
	pergeseran merah: GetReservedNodeExchangeOfferings
	pergeseran merah: GetResourcePolicy
	pergeseran merah: ListRecommendations
	pergeseran merah: ModifyAquaConfiguration
	pergeseran merah: ModifyAuthenticationProfile
	pergeseran merah: ModifyCluster
	pergeseran merah: ModifyClusterDbRevision
	pergeseran merah: ModifyClusterIamRoles
	pergeseran merah: ModifyClusterMaintenance
	pergeseran merah: ModifyClusterParameterGroup

Awalan layanan	Tindakan
	pergeseran merah: ModifyClusterSnapshot
	pergeseran merah: ModifyClusterSnapshotSchedule
	pergeseran merah: ModifyClusterSubnetGroup
	pergeseran merah: ModifyCustomDomainAssociation
	pergeseran merah: ModifyEndpointAccess
	pergeseran merah: ModifyEventSubscription
	pergeseran merah: ModifyRedshiftIdcApplication
	pergeseran merah: ModifyScheduledAction
	pergeseran merah: ModifySnapshotCopyRetentionPeriod
	pergeseran merah: ModifySnapshotSchedule
	pergeseran merah: ModifyUsageLimit
	pergeseran merah: PauseCluster
	pergeseran merah: PurchaseReservedNodeOffering
	pergeseran merah: PutResourcePolicy
	pergeseran merah: RebootCluster
	pergeseran merah: RejectDataShare
	pergeseran merah: ResetClusterParameterGroup
	pergeseran merah: ResizeCluster
	pergeseran merah: RestoreFromClusterSnapshot
	pergeseran merah: RestoreTableFromClusterSnapshot
	pergeseran merah: ResumeCluster

Awalan layanan	Tindakan
	pergeseran merah: RevokeClusterSecurityGroupIngress pergeseran merah: RevokeEndpointAccess pergeseran merah: RevokeSnapshotAccess pergeseran merah: RotateEncryptionKey pergeseran merah: UpdatePartnerStatus
data pergeseran merah	data pergeseran merah: BatchExecuteStatement data pergeseran merah: CancelStatement data pergeseran merah: DescribeStatement data pergeseran merah: DescribeTable data pergeseran merah: ExecuteStatement data pergeseran merah: GetStatementResult data pergeseran merah: ListDatabases data pergeseran merah: ListSchemas data pergeseran merah: ListStatements data pergeseran merah: ListTables

Awalan layanan	Tindakan
ruang refaktor	ruang refaktor: CreateApplication
	ruang refaktor: CreateEnvironment
	ruang refaktor: CreateRoute
	ruang refaktor: CreateService
	ruang refaktor: DeleteApplication
	ruang refaktor: DeleteEnvironment
	ruang refaktor: DeleteResourcePolicy
	ruang refaktor: DeleteRoute
	ruang refaktor: DeleteService
	ruang refaktor: GetApplication
	ruang refaktor: GetEnvironment
	ruang refaktor: GetResourcePolicy
	ruang refaktor: GetRoute
	ruang refaktor: GetService
	ruang refaktor: ListApplications
	ruang refaktor: ListEnvironments
	ruang refaktor: ListEnvironmentVpcs
	ruang refaktor: ListRoutes
	ruang refaktor: ListServices
	ruang refaktor: PutResourcePolicy
	ruang refaktor: UpdateRoute

Awalan layanan	Tindakan
rekognisi	Rekognisi: AssociateFaces
	Rekognisi: CompareFaces
	Rekognisi: CopyProjectVersion
	Rekognisi: CreateCollection
	Rekognisi: CreateDataset
	Rekognisi: CreateFaceLivenessSession
	Rekognisi: CreateProject
	Rekognisi: CreateProjectVersion
	Rekognisi: CreateStreamProcessor
	Rekognisi: CreateUser
	Rekognisi: DeleteCollection
	Rekognisi: DeleteDataset
	Rekognisi: DeleteFaces
	Rekognisi: DeleteProject
	Rekognisi: DeleteProjectPolicy
	Rekognisi: DeleteProjectVersion
	Rekognisi: DeleteStreamProcessor
	Rekognisi: DeleteUser
	Rekognisi: DescribeCollection
	Rekognisi: DescribeDataset
	Rekognisi: DescribeProjects

Awalan layanan	Tindakan
	Rekognisi: DescribeProjectVersions
	Rekognisi: DescribeStreamProcessor
	Rekognisi: DetectCustomLabels
	Rekognisi: DetectFaces
	Rekognisi: DetectLabels
	Rekognisi: DetectModerationLabels
	Rekognisi: DetectProtectiveEquipment
	Rekognisi: DetectText
	Rekognisi: DisassociateFaces
	Rekognisi: DistributeDatasetEntries
	Rekognisi: GetCelebrityInfo
	Rekognisi: GetCelebrityRecognition
	Rekognisi: GetContentModeration
	Rekognisi: GetFaceDetection
	Rekognisi: GetFaceLivenessSessionResults
	Rekognisi: GetFaceSearch
	Rekognisi: GetLabelDetection
	Rekognisi: GetMediaAnalysisJob
	Rekognisi: GetPersonTracking
	Rekognisi: GetSegmentDetection
	Rekognisi: GetTextDetection

Awalan layanan	Tindakan
	Rekognisi: IndexFaces
	Rekognisi: ListCollections
	Rekognisi: ListDatasetEntries
	Rekognisi: ListDatasetLabels
	Rekognisi: ListFaces
	Rekognisi: ListMediaAnalysisJobs
	Rekognisi: ListProjectPolicies
	Rekognisi: ListStreamProcessors
	Rekognisi: ListUsers
	Rekognisi: PutProjectPolicy
	Rekognisi: RecognizeCelebrities
	Rekognisi: SearchFaces
	Rekognisi: SearchFacesByImage
	Rekognisi: SearchUsers
	Rekognisi: SearchUsersByImage
	Rekognisi: StartCelebrityRecognition
	Rekognisi: StartContentModeration
	Rekognisi: StartFaceDetection
	Rekognisi: StartFaceLivenessSession
	Rekognisi: StartFaceSearch
	Rekognisi: StartLabelDetection

Awalan layanan	Tindakan
	Rekognisi: StartMediaAnalysisJob Rekognisi: StartPersonTracking Rekognisi: StartProjectVersion Rekognisi: StartSegmentDetection Rekognisi: StartStreamProcessor Rekognisi: StartTextDetection Rekognisi: StopProjectVersion Rekognisi: StopStreamProcessor Rekognisi: UpdateDatasetEntries Rekognisi: UpdateStreamProcessor

Awalan layanan	Tindakan
resiliencehub	resiliencehub: AcceptResourceGroupingRecommendations
	resiliencehub: AddDraftAppVersionResourceMappings
	resiliencehub: BatchUpdateRecommendationStatus
	resiliencehub: CreateApp
	resiliencehub: CreateAppVersionAppComponent
	resiliencehub: CreateAppVersionResource
	resiliencehub: CreateRecommendationTemplate
	resiliencehub: CreateResiliencyPolicy
	resiliencehub: DeleteApp
	resiliencehub: DeleteAppAssessment
	resiliencehub: DeleteAppInputSource
	resiliencehub: DeleteAppVersionAppComponent
	resiliencehub: DeleteAppVersionResource
	resiliencehub: DeleteRecommendationTemplate
	resiliencehub: DeleteResiliencyPolicy
	resiliencehub: DescribeApp
	resiliencehub: DescribeAppAssessment
	resiliencehub: DescribeAppVersion
	resiliencehub: DescribeAppVersionAppComponent
	resiliencehub: DescribeAppVersionResource
	resiliencehub: DescribeAppVersionResourcesResolutionStatus

Awalan layanan	Tindakan
	resiliencehub: DescribeAppVersionTemplate
	resiliencehub: DescribeDraftAppVersionResourcesImportStatus
	resiliencehub: DescribeResiliencyPolicy
	resiliencehub: DescribeResourceGroupingRecommendationTask
	resiliencehub: ImportResourcesToDraftAppVersion
	resiliencehub: ListAlarmRecommendations
	resiliencehub: ListAppAssessmentComplianceDrifts
	resiliencehub: ListAppAssessmentResourceDrifts
	resiliencehub: ListAppAssessments
	resiliencehub: ListAppComponentCompliances
	resiliencehub: ListAppComponentRecommendations
	resiliencehub: ListAppInputSources
	resiliencehub: ListApps
	resiliencehub: ListAppVersionAppComponents
	resiliencehub: ListAppVersionResourceMappings
	resiliencehub: ListAppVersionResources
	resiliencehub: ListAppVersions
	resiliencehub: ListRecommendationTemplates
	resiliencehub: ListResiliencyPolicies
	resiliencehub: ListResourceGroupingRecommendations
	resiliencehub: ListSopRecommendations

Awalan layanan	Tindakan
	resiliencehub: ListSuggestedResiliencyPolicies
	resiliencehub: ListTestRecommendations
	resiliencehub: ListUnsupportedAppVersionResources
	resiliencehub: PublishAppVersion
	resiliencehub: PutDraftAppVersionTemplate
	resiliencehub: RejectResourceGroupingRecommendations
	resiliencehub: RemoveDraftAppVersionResourceMappings
	resiliencehub: ResolveAppVersionResources
	resiliencehub: StartAppAssessment
	resiliencehub: StartResourceGroupingRecommendationTask
	resiliencehub: UpdateApp
	resiliencehub: UpdateAppVersion
	resiliencehub: UpdateAppVersionAppComponent
	resiliencehub: UpdateAppVersionResource
	resiliencehub: UpdateResiliencyPolicy

Awalan layanan	Tindakan
sumber daya-penjelajah-2	sumber daya-penjelajah-2: AssociateDefaultView sumber daya-penjelajah-2: BatchGetView sumber daya-penjelajah-2: CreateIndex sumber daya-penjelajah-2: CreateView sumber daya-penjelajah-2: DeleteIndex sumber daya-penjelajah-2: DeleteView sumber daya-penjelajah-2: DisassociateDefaultView sumber daya-penjelajah-2: GetAccountLevelServiceConfiguration sumber daya-penjelajah-2: GetDefaultView sumber daya-penjelajah-2: GetIndex sumber daya-penjelajah-2: ListIndexes sumber daya-penjelajah-2: ListIndexesForMembers sumber daya-penjelajah-2: ListSupportedResourceTypes sumber daya-penjelajah-2: ListViews Resource-Explorer-2:Search sumber daya-penjelajah-2: UpdateIndexType sumber daya-penjelajah-2: UpdateView

Awalan layanan	Tindakan
kelompok sumber daya	kelompok sumber daya: CreateGroup
	kelompok sumber daya: DeleteGroup
	kelompok sumber daya: GetAccountSettings
	kelompok sumber daya: GetGroup
	kelompok sumber daya: GetGroupConfiguration
	kelompok sumber daya: GetGroupQuery
	kelompok sumber daya: GroupResources
	kelompok sumber daya: ListGroupResources
	kelompok sumber daya: ListGroups
	kelompok sumber daya: PutGroupConfiguration
	kelompok sumber daya: SearchResources
	kelompok sumber daya: UngroupResources
	kelompok sumber daya: UpdateAccountSettings
	kelompok sumber daya: UpdateGroup
	kelompok sumber daya: UpdateGroupQuery

Awalan layanan	Tindakan
pembuat robomak	pembuat robomak: BatchDeleteWorlds pembuat robomak: BatchDescribeSimulationJob pembuat robomak: CancelDeploymentJob pembuat robomak: CancelSimulationJob pembuat robomak: CancelSimulationJobBatch pembuat robomak: CancelWorldExportJob pembuat robomak: CancelWorldGenerationJob pembuat robomak: CreateDeploymentJob pembuat robomak: CreateFleet pembuat robomak: CreateRobot pembuat robomak: CreateRobotApplication pembuat robomak: CreateRobotApplicationVersion pembuat robomak: CreateSimulationApplication pembuat robomak: CreateSimulationApplicationVersion pembuat robomak: CreateSimulationJob pembuat robomak: CreateWorldExportJob pembuat robomak: CreateWorldGenerationJob pembuat robomak: CreateWorldTemplate pembuat robomak: DeleteFleet pembuat robomak: DeleteRobot pembuat robomak: DeleteRobotApplication

Awalan layanan	Tindakan
	pembuat robomak: DeleteSimulationApplication
	pembuat robomak: DeleteWorldTemplate
	pembuat robomak: DeregisterRobot
	pembuat robomak: DescribeDeploymentJob
	pembuat robomak: DescribeFleet
	pembuat robomak: DescribeRobot
	pembuat robomak: DescribeRobotApplication
	pembuat robomak: DescribeSimulationApplication
	pembuat robomak: DescribeSimulationJob
	pembuat robomak: DescribeSimulationJobBatch
	pembuat robomak: DescribeWorld
	pembuat robomak: DescribeWorldExportJob
	pembuat robomak: DescribeWorldGenerationJob
	pembuat robomak: DescribeWorldTemplate
	pembuat robomak: GetWorldTemplateBody
	pembuat robomak: ListDeploymentJobs
	pembuat robomak: ListFleets
	pembuat robomak: ListRobotApplications
	pembuat robomak: ListRobots
	pembuat robomak: ListSimulationApplications
	pembuat robomak: ListSimulationJobBatches

Awalan layanan	Tindakan
	pembuat robomak: ListSimulationJobs pembuat robomak: ListWorldExportJobs pembuat robomak: ListWorldGenerationJobs pembuat robomak: ListWorlds pembuat robomak: ListWorldTemplates pembuat robomak: RegisterRobot pembuat robomak: RestartSimulationJob pembuat robomak: StartSimulationJobBatch pembuat robomak: SyncDeploymentJob pembuat robomak: UpdateRobotApplication pembuat robomak: UpdateSimulationApplication pembuat robomak: UpdateWorldTemplate

Awalan layanan	Tindakan
peranandi mana saja	peranandi mana saja: CreateProfile
	peranandi mana saja: CreateTrustAnchor
	peranandi mana saja: DeleteAttributeMapping
	peranandi mana saja: DeleteCrl
	peranandi mana saja: DeleteProfile
	peranandi mana saja: DeleteTrustAnchor
	peranandi mana saja: DisableCrl
	peranandi mana saja: DisableProfile
	peranandi mana saja: DisableTrustAnchor
	peranandi mana saja: EnableCrl
	peranandi mana saja: EnableProfile
	peranandi mana saja: EnableTrustAnchor
	peranandi mana saja: GetCrl
	peranandi mana saja: GetProfile
	peranandi mana saja: GetSubject
	peranandi mana saja: GetTrustAnchor
	peranandi mana saja: ImportCrl
	peranandi mana saja: ListCrls
	peranandi mana saja: ListProfiles
	peranandi mana saja: ListSubjects
	peranandi mana saja: ListTrustAnchors

Awalan layanan	Tindakan
	peranandi mana saja: PutAttributeMapping
	peranandi mana saja: PutNotificationSettings
	peranandi mana saja: ResetNotificationSettings
	peranandi mana saja: UpdateCrl
	peranandi mana saja: UpdateProfile
	peranandi mana saja: UpdateTrustAnchor

Awalan layanan	Tindakan
route53	route53: ActivateKeySigningKey Route53:a ssociateVPCWith HostedZone route53: ChangeCidrCollection route53: ChangeResourceRecordSets route53: CreateCidrCollection route53: CreateHealthCheck route53: CreateHostedZone route53: CreateKeySigningKey route53: CreateQueryLoggingConfig route53: CreateReusableDelegationSet route53: CreateTrafficPolicy route53: CreateTrafficPolicyInstance route53: CreateTrafficPolicyVersion Otorisasi Route53reateVPCAssociation: C route53: DeactivateKeySigningKey route53: DeleteCidrCollection route53: DeleteHealthCheck route53: DeleteHostedZone route53: DeleteKeySigningKey route53: DeleteQueryLoggingConfig route53: DeleteReusableDelegationSet

Awalan layanan	Tindakan
	route53: DeleteTrafficPolicy route53: DeleteTrafficPolicyInstance Otorisasi Route53eleteVPCAssociation: D route53: DisableHostedZone DNSSEC Route53:d isassociateVPCFrom HostedZone route53: EnableHostedZone DNSSEC route53: GetAccountLimit route53: GetChange route53: GetCheckerIpRanges Route53:Dapatkan DNSSEC route53: GetGeoLocation route53: GetHealthCheck route53: GetHealthCheckCount route53: GetHealthCheckLastFailureReason route53: GetHealthCheckStatus route53: GetHostedZone route53: GetHostedZoneCount route53: GetHostedZoneLimit route53: GetQueryLoggingConfig route53: GetReusableDelegationSet route53: GetReusableDelegationSetLimit

Awalan layanan	Tindakan
	route53: GetTrafficPolicy route53: GetTrafficPolicyInstance route53: GetTrafficPolicyInstanceCount route53: ListCidrBlocks route53: ListCidrCollections route53: ListCidrLocations route53: ListGeoLocations route53: ListHealthChecks route53: ListHostedZones route53: ListHostedZonesByName route53: ListHostedZonesBy VPC route53: ListQueryLoggingConfigs route53: ListResourceRecordSets route53: ListReusableDelegationSets route53: ListTrafficPolicies route53: ListTrafficPolicyInstances route53: ListTrafficPolicyInstancesByHostedZone route53: ListTrafficPolicyInstancesByPolicy route53: ListTrafficPolicyVersions Otorisasi Route53istVPCAssociation: L Route53: t estDNSAnswer

Awalan layanan	Tindakan
	route53: UpdateHealthCheck route53: UpdateHostedZoneComment route53: UpdateTrafficPolicyComment route53: UpdateTrafficPolicyInstance

Awalan layanan	Tindakan
rute53- recovery-control-config	route53-: recovery-control-config CreateCluster
	route53-: recovery-control-config CreateControlPanel
	route53-: recovery-control-config CreateRoutingControl
	route53-: recovery-control-config CreateSafetyRule
	route53-: recovery-control-config DeleteCluster
	route53-: recovery-control-config DeleteControlPanel
	route53-: recovery-control-config DeleteRoutingControl
	route53-: recovery-control-config DeleteSafetyRule
	route53-: recovery-control-config DescribeCluster
	route53-: recovery-control-config DescribeControlPanel
	route53-: recovery-control-config DescribeRoutingControl
	route53-: recovery-control-config DescribeSafetyRule
	route53-: recovery-control-config GetResourcePolicy
	rute53-: recovery-control-config 53 ListAssociatedRoute HealthChecks
	route53-: recovery-control-config ListClusters
	route53-: recovery-control-config ListControlPanels
	route53-: recovery-control-config ListRoutingControls
	route53-: recovery-control-config ListSafetyRules
	route53-: recovery-control-config UpdateControlPanel
	route53-: recovery-control-config UpdateRoutingControl

Awalan layanan	Tindakan
	route53-: recovery-control-config UpdateSafetyRule

Awalan layanan	Tindakan
route53-pemulihan-kesiapan	route53-pemulihan-kesiapan: CreateCell route53-pemulihan-kesiapan: CreateCrossAccountAuthorization route53-pemulihan-kesiapan: CreateReadinessCheck route53-pemulihan-kesiapan: CreateRecoveryGroup route53-pemulihan-kesiapan: CreateResourceSet route53-pemulihan-kesiapan: DeleteCell route53-pemulihan-kesiapan: DeleteCrossAccountAuthorization route53-pemulihan-kesiapan: DeleteReadinessCheck route53-pemulihan-kesiapan: DeleteRecoveryGroup route53-pemulihan-kesiapan: DeleteResourceSet route53-pemulihan-kesiapan: GetArchitectureRecommendations route53-pemulihan-kesiapan: GetCell route53-pemulihan-kesiapan: GetCellReadinessSummary route53-pemulihan-kesiapan: GetReadinessCheck route53-pemulihan-kesiapan: GetReadinessCheckResourceStatus route53-pemulihan-kesiapan: GetReadinessCheckStatus route53-pemulihan-kesiapan: GetRecoveryGroup route53-pemulihan-kesiapan: GetRecoveryGroupReadinessSummary route53-pemulihan-kesiapan: GetResourceSet route53-pemulihan-kesiapan: ListCells

Awalan layanan	Tindakan
	route53-pemulihan-kesiapan: ListCrossAccountAuthorizations route53-pemulihan-kesiapan: ListReadinessChecks route53-pemulihan-kesiapan: ListRecoveryGroups route53-pemulihan-kesiapan: ListResourceSets route53-pemulihan-kesiapan: ListRules route53-pemulihan-kesiapan: UpdateCell route53-pemulihan-kesiapan: UpdateReadinessCheck route53-pemulihan-kesiapan: UpdateRecoveryGroup route53-pemulihan-kesiapan: UpdateResourceSet

Awalan layanan	Tindakan
route53resolver	route53resolver: AssociateFirewallRuleGroup
	route53resolver: AssociateResolverEndpointIpAddress
	route53resolver: AssociateResolverQueryLogConfig
	route53resolver: AssociateResolverRule
	route53resolver: CreateFirewallDomainList
	route53resolver: CreateFirewallRule
	route53resolver: CreateFirewallRuleGroup
	route53resolver: CreateResolverEndpoint
	route53resolver: CreateResolverQueryLogConfig
	route53resolver: CreateResolverRule
	route53resolver: DeleteFirewallDomainList
	route53resolver: DeleteFirewallRule
	route53resolver: DeleteFirewallRuleGroup
	route53resolver: DeleteOutpostResolver
	route53resolver: DeleteResolverEndpoint
	route53resolver: DeleteResolverQueryLogConfig
	route53resolver: DeleteResolverRule
	route53resolver: DisassociateFirewallRuleGroup
	route53resolver: DisassociateResolverEndpointIpAddress
	route53resolver: DisassociateResolverQueryLogConfig
	route53resolver: DisassociateResolverRule

Awalan layanan	Tindakan
	route53resolver: GetFirewallConfig
	route53resolver: GetFirewallDomainList
	route53resolver: GetFirewallRuleGroup
	route53resolver: GetFirewallRuleGroupAssociation
	route53resolver: GetFirewallRuleGroupPolicy
	route53resolver: GetOutpostResolver
	route53resolver: GetResolverConfig
	route53resolver: GetResolverDnssecConfig
	route53resolver: GetResolverEndpoint
	route53resolver: GetResolverQueryLogConfig
	route53resolver: GetResolverQueryLogConfigAssociation
	route53resolver: GetResolverQueryLogConfigPolicy
	route53resolver: GetResolverRule
	route53resolver: GetResolverRuleAssociation
	route53resolver: GetResolverRulePolicy
	route53resolver: ImportFirewallDomains
	route53resolver: ListFirewallConfigs
	route53resolver: ListFirewallDomainLists
	route53resolver: ListFirewallDomains
	route53resolver: ListFirewallRuleGroupAssociations
	route53resolver: ListFirewallRuleGroups

Awalan layanan	Tindakan
	route53resolver: ListFirewallRules
	route53resolver: ListOutpostResolvers
	route53resolver: ListResolverConfigs
	route53resolver: ListResolverDnssecConfigs
	route53resolver: ListResolverEndpointIpAddresses
	route53resolver: ListResolverEndpoints
	route53resolver: ListResolverQueryLogConfigAssociations
	route53resolver: ListResolverQueryLogConfigs
	route53resolver: ListResolverRuleAssociations
	route53resolver: ListResolverRules
	route53resolver: PutFirewallRuleGroupPolicy
	route53resolver: PutResolverQueryLogConfigPolicy
	route53resolver: UpdateFirewallConfig
	route53resolver: UpdateFirewallDomains
	route53resolver: UpdateFirewallRule
	route53resolver: UpdateFirewallRuleGroupAssociation
	route53resolver: UpdateOutpostResolver
	route53resolver: UpdateResolverConfig
	route53resolver: UpdateResolverDnssecConfig
	route53resolver: UpdateResolverEndpoint
	route53resolver: UpdateResolverRule

Awalan layanan	Tindakan
rum	rum: BatchCreateRumMetricDefinitions rum: BatchDeleteRumMetricDefinitions rum: BatchGetRumMetricDefinitions rum: CreateAppMonitor rum: DeleteAppMonitor rum: DeleteRumMetricsDestination rum: GetAppMonitor rum: GetAppMonitorData rum: ListAppMonitors rum: ListRumMetricsDestinations rum: PutRumMetricsDestination rum: UpdateAppMonitor rum: UpdateRumMetricDefinition

Awalan layanan	Tindakan
s3	s3: AssociateAccessGrantsIdentityCenter
	s3: CreateAccessGrant
	s3: CreateAccessGrantsInstance
	s3: CreateAccessGrantsLocation
	s3: CreateAccessPoint
	s3: CreateAccessPointForObjectLambda
	s3: CreateBucket
	s3: CreateJob
	s3: CreateMultiRegionAccessPoint
	s3: DeleteAccessGrant
	s3: DeleteAccessGrantsInstance
	s3: DeleteAccessGrantsInstanceResourcePolicy
	s3: DeleteAccessGrantsLocation
	s3: DeleteAccessPoint
	s3: DeleteAccessPointForObjectLambda
	s3: DeleteAccessPointPolicy
	s3: DeleteAccessPointPolicyForObjectLambda
	s3: PutAccountPublicAccessBlock
	s3: DeleteBucket
	s3: PutAnalyticsConfiguration
	s3: PutBucket CORS

Awalan layanan	Tindakan
	s3: PutEncryptionConfiguration
	s3: PutIntelligentTieringConfiguration
	s3: PutInventoryConfiguration
	s3: PutLifecycleConfiguration
	s3: PutMetricsConfiguration
	s3: PutBucketOwnershipControls
	s3: DeleteBucketPolicy
	s3: PutBucketPublicAccessBlock
	s3: PutReplicationConfiguration
	s3: DeleteBucketWebsite
	s3: DeleteMultiRegionAccessPoint
	s3: DeleteStorageLensConfiguration
	s3: DescribeJob
	s3: DescribeMultiRegionAccessPointOperation
	s3: DissociateAccessGrantsIdentityCenter
	s3: GetAccelerateConfiguration
	s3: GetAccessGrant
	s3: GetAccessGrantsInstance
	s3: GetAccessGrantsInstanceForPrefix
	s3: GetAccessGrantsInstanceResourcePolicy
	s3: GetAccessGrantsLocation

Awalan layanan	Tindakan
	s3: GetAccessPoint
	s3: GetAccessPointConfigurationForObjectLambda
	s3: GetAccessPointForObjectLambda
	s3: GetAccessPointPolicy
	s3: GetAccessPointPolicyForObjectLambda
	s3: GetAccessPointPolicyStatus
	s3: GetAccessPointPolicyStatusForObjectLambda
	s3: GetAccountPublicAccessBlock
	s3: GetBucketAcl
	s3: GetAnalyticsConfiguration
	s3: GetBucket CORS
	s3: GetEncryptionConfiguration
	s3: GetIntelligentTieringConfiguration
	s3: GetInventoryConfiguration
	s3: GetLifecycleConfiguration
	s3: GetBucketLocation
	s3: GetBucketLogging
	s3: GetMetricsConfiguration
	s3: GetBucketNotification
	s3: GetBucketObjectLockConfiguration
	s3: GetBucketOwnershipControls

Awalan layanan	Tindakan
	s3: GetBucketPolicy
	s3: GetBucketPolicyStatus
	s3: GetBucketPublicAccessBlock
	s3: GetReplicationConfiguration
	s3: GetBucketRequestPayment
	s3: GetBucketVersioning
	s3: GetBucketWebsite
	s3: GetDataAccess
	s3: GetMultiRegionAccessPoint
	s3: GetMultiRegionAccessPointPolicy
	s3: GetMultiRegionAccessPointPolicyStatus
	s3: GetMultiRegionAccessPointRoutes
	s3: GetObjectAttributes
	s3: GetStorageLensConfiguration
	s3: GetStorageLensDashboard
	s3: ListAccessGrants
	s3: ListAccessGrantsInstances
	s3: ListAccessGrantsLocations
	s3: ListAccessPoints
	s3: ListAccessPointsForObjectLambda
	s3: ListAllMyBuckets

Awalan layanan	Tindakan
	s3: ListJobs
	s3: ListBucketMultipartUploads
	s3: ListMultiRegionAccessPoints
	s3: ListStorageLensConfigurations
	s3: PutAccelerateConfiguration
	s3: PutAccessGrantsInstanceResourcePolicy
	s3: PutAccessPointConfigurationForObjectLambda
	s3: PutAccessPointPolicy
	s3: PutAccessPointPolicyForObjectLambda
	s3: PutAccountPublicAccessBlock
	s3: PutBucketAcl
	s3: PutAnalyticsConfiguration
	s3: PutBucket CORS
	s3: PutEncryptionConfiguration
	s3: PutIntelligentTieringConfiguration
	s3: PutInventoryConfiguration
	s3: PutLifecycleConfiguration
	s3: PutBucketLogging
	s3: PutMetricsConfiguration
	s3: PutBucketNotification
	s3: PutBucketObjectLockConfiguration

Awalan layanan	Tindakan
	s3: PutBucketOwnershipControls s3: PutBucketPolicy s3: PutBucketPublicAccessBlock s3: PutReplicationConfiguration s3: PutBucketRequestPayment s3: PutBucketVersioning s3: PutBucketWebsite s3: PutMultiRegionAccessPointPolicy s3: PutStorageLensConfiguration s3: SubmitMultiRegionAccessPointRoutes s3: UpdateAccessGrantsLocation s3: UpdateJobPriority s3: UpdateJobStatus
pos terdepan s3	pos terdepan s3: CreateEndpoint pos terdepan s3: DeleteEndpoint pos terdepan s3: ListEndpoints s3-pos terdepan: S3 ListOutpostsWith pos terdepan s3: ListSharedEndpoints

Awalan layanan	Tindakan
sagemaker-geospasial	sagemaker-geospasial: DeleteEarthObservationJob
	sagemaker-geospasial: DeleteVectorEnrichmentJob
	sagemaker-geospasial: ExportEarthObservationJob
	sagemaker-geospasial: ExportVectorEnrichmentJob
	sagemaker-geospasial: GetEarthObservationJob
	sagemaker-geospasial: GetRasterDataCollection
	sagemaker-geospasial: GetTile
	sagemaker-geospasial: GetVectorEnrichmentJob
	sagemaker-geospasial: ListEarthObservationJobs
	sagemaker-geospasial: ListRasterDataCollections
	sagemaker-geospasial: ListVectorEnrichmentJobs
	sagemaker-geospasial: SearchRasterDataCollection
	sagemaker-geospasial: StartEarthObservationJob
	sagemaker-geospasial: StartVectorEnrichmentJob
	sagemaker-geospasial: StopEarthObservationJob
	sagemaker-geospasial: StopVectorEnrichmentJob

Awalan layanan	Tindakan
savingsplans	savingsplans: CreateSavingsPlan savingsplans: DeleteQueuedSavingsPlan savingsplans: DescribeSavingsPlanRates savingsplans: DescribeSavingsPlans savingsplans: DescribeSavingsPlansOfferingRates savingsplans: DescribeSavingsPlansOfferings savingsplans: ReturnSavingsPlan

Awalan layanan	Tindakan
skema	skema: CreateDiscoverer
	skema: CreateRegistry
	skema: CreateSchema
	skema: DeleteDiscoverer
	skema: DeleteRegistry
	skema: DeleteResourcePolicy
	skema: DeleteSchema
	skema: DeleteSchemaVersion
	skema: DescribeCodeBinding
	skema: DescribeDiscoverer
	skema: DescribeRegistry
	skema: DescribeSchema
	skema: ExportSchema
	skema: GetCodeBindingSource
	skema: GetDiscoveredSchema
	skema: GetResourcePolicy
	skema: ListDiscoverers
	skema: ListRegistries
	skema: ListSchemas
	skema: ListSchemaVersions
	skema: PutCodeBinding

Awalan layanan	Tindakan
	skema: PutResourcePolicy skema: SearchSchemas skema: StartDiscoverer skema: StopDiscoverer skema: UpdateDiscoverer skema: UpdateRegistry skema: UpdateSchema
sdb	sdb: CreateDomain sdb: DeleteDomain sdb: DomainMetadata sdb: ListDomains

Awalan layanan	Tindakan
manajer rahasia	manajer rahasia: CancelRotateSecret
	manajer rahasia: CreateSecret
	manajer rahasia: DeleteResourcePolicy
	manajer rahasia: DeleteSecret
	manajer rahasia: DescribeSecret
	manajer rahasia: GetRandomPassword
	manajer rahasia: GetResourcePolicy
	manajer rahasia: GetSecretValue
	manajer rahasia: ListSecrets
	manajer rahasia: ListSecretVersionIds
	manajer rahasia: PutResourcePolicy
	manajer rahasia: PutSecretValue
	manajer rahasia: RemoveRegionsFromReplication
	manajer rahasia: ReplicateSecretToRegions
	manajer rahasia: RestoreSecret
	manajer rahasia: RotateSecret
	manajer rahasia: StopReplicationToReplica
	manajer rahasia: UpdateSecret
	manajer rahasia: ValidateResourcePolicy

Awalan layanan	Tindakan
securityhub	securityhub: AcceptAdministratorInvitation
	securityhub: AcceptInvitation
	securityhub: BatchDeleteAutomationRules
	securityhub: BatchDisableStandards
	securityhub: BatchEnableStandards
	securityhub: BatchGetAutomationRules
	securityhub: BatchGetConfigurationPolicyAssociations
	securityhub: BatchGetSecurityControls
	securityhub: BatchGetStandardsControlAssociations
	securityhub: BatchImportFindings
	securityhub: BatchUpdateAutomationRules
	securityhub: BatchUpdateFindings
	securityhub: BatchUpdateStandardsControlAssociations
	securityhub: CreateActionTarget
	securityhub: CreateAutomationRule
	securityhub: CreateConfigurationPolicy
	securityhub: CreateFindingAggregator
	securityhub: CreateInsight
	securityhub: CreateMembers
	securityhub: DeclineInvitations
	securityhub: DeleteActionTarget

Awalan layanan	Tindakan
	securityhub: DeleteConfigurationPolicy securityhub: DeleteFindingAggregator securityhub: DeleteInsight securityhub: DeleteInvitations securityhub: DeleteMembers securityhub: DescribeActionTargets securityhub: DescribeHub securityhub: DescribeOrganizationConfiguration securityhub: DescribeProducts securityhub: DescribeStandards securityhub: DisableImportFindingsForProduct securityhub: DisableOrganizationAdminAccount securityhub: DisableSecurityHub securityhub: DisassociateFromAdministratorAccount securityhub: DisassociateFromMasterAccount securityhub: DisassociateMembers securityhub: EnableImportFindingsForProduct securityhub: EnableOrganizationAdminAccount securityhub: EnableSecurityHub securityhub: GetAdministratorAccount securityhub: GetConfigurationPolicy

Awalan layanan	Tindakan
	securityhub: GetConfigurationPolicyAssociation securityhub: GetEnabledStandards securityhub: GetFindingAggregator securityhub: GetFindingHistory securityhub: GetFindings securityhub: GetInsightResults securityhub: GetInsights securityhub: GetInvitationsCount securityhub: GetMasterAccount securityhub: GetMembers securityhub: GetSecurityControlDefinition securityhub: InviteMembers securityhub: ListAutomationRules securityhub: ListConfigurationPolicies securityhub: ListConfigurationPolicyAssociations securityhub: ListEnabledProductsForImport securityhub: ListFindingAggregators securityhub: ListInvitations securityhub: ListMembers securityhub: ListOrganizationAdminAccounts securityhub: ListSecurityControlDefinitions

Awalan layanan	Tindakan
	securityhub: ListStandardsControlAssociations securityhub: StartConfigurationPolicyAssociation securityhub: StartConfigurationPolicyDisassociation securityhub: UpdateActionTarget securityhub: UpdateConfigurationPolicy securityhub: UpdateFindingAggregator securityhub: UpdateFindings securityhub: UpdateInsight securityhub: UpdateOrganizationConfiguration securityhub: UpdateSecurityControl securityhub: UpdateSecurityHubConfiguration

Awalan layanan	Tindakan
Securitylake	Securitylake: CreateAwsLogSource
	Securitylake: CreateCustomLogSource
	Securitylake: CreateDataLakeExceptionSubscription
	Securitylake: CreateDataLakeOrganizationConfiguration
	Securitylake: CreateSubscriber
	Securitylake: CreateSubscriberNotification
	Securitylake: DeleteAwsLogSource
	Securitylake: DeleteCustomLogSource
	Securitylake: DeleteDataLakeExceptionSubscription
	Securitylake: DeleteDataLakeOrganizationConfiguration
	Securitylake: DeleteSubscriber
	Securitylake: DeleteSubscriberNotification
	Securitylake: DeregisterDataLakeDelegatedAdministrator
	Securitylake: GetDataLakeExceptionSubscription
	Securitylake: GetDataLakeOrganizationConfiguration
	Securitylake: GetDataLakeSources
	Securitylake: GetSubscriber
	Securitylake: ListDataLakes
	Securitylake: ListLogSources
	Securitylake: ListSubscribers
	Securitylake: RegisterDataLakeDelegatedAdministrator

Awalan layanan	Tindakan
	Securitylake: UpdateDataLakeExceptionSubscription Securitylake: UpdateSubscriber Securitylake: UpdateSubscriberNotification
repo tanpa server	repo tanpa server: CreateApplication repo tanpa server: CreateApplicationVersion repo tanpa server: CreateCloudFormationChangeSet repo tanpa server: CreateCloudFormationTemplate repo tanpa server: DeleteApplication repo tanpa server: GetApplication repo tanpa server: GetApplicationPolicy repo tanpa server: GetCloudFormationTemplate repo tanpa server: ListApplicationDependencies repo tanpa server: ListApplications repo tanpa server: ListApplicationVersions repo tanpa server: PutApplicationPolicy repo tanpa server: UnshareApplication repo tanpa server: UpdateApplication

Awalan layanan	Tindakan
servicecatalog	servicecatalog: AcceptPortfolioShare
	servicecatalog: AssociateBudgetWithResource
	servicecatalog: AssociatePrincipalWithPortfolio
	servicecatalog: AssociateProductWithPortfolio
	servicecatalog: AssociateServiceActionWithProvisioningArtifact
	servicecatalog: BatchAssociateServiceActionWithProvisioningArtifact
	servicecatalog: BatchDisassociateServiceActionFromProvisioningArtifact
	servicecatalog: CopyProduct
	servicecatalog: CreateAttributeGroup
	servicecatalog: CreateConstraint
	servicecatalog: CreatePortfolio
	servicecatalog: CreatePortfolioShare
	servicecatalog: CreateProduct
	servicecatalog: CreateProvisionedProductPlan
	servicecatalog: CreateProvisioningArtifact
	servicecatalog: CreateServiceAction
	servicecatalog: DeleteAttributeGroup
	servicecatalog: DeleteConstraint
	servicecatalog: DeletePortfolio
	servicecatalog: DeletePortfolioShare

Awalan layanan	Tindakan
	servicecatalog: DeleteProduct
	servicecatalog: DeleteProvisionedProductPlan
	servicecatalog: DeleteProvisioningArtifact
	servicecatalog: DeleteServiceAction
	servicecatalog: DescribeConstraint
	servicecatalog: DescribeCopyProductStatus
	servicecatalog: DescribePortfolio
	servicecatalog: DescribePortfolioShares
	servicecatalog: DescribePortfolioShareStatus
	servicecatalog: DescribeProduct
	servicecatalog: DescribeProductAsAdmin
	servicecatalog: DescribeProductView
	servicecatalog: DescribeProvisionedProduct
	servicecatalog: DescribeProvisionedProductPlan
	servicecatalog: DescribeProvisioningArtifact
	servicecatalog: DescribeProvisioningParameters
	servicecatalog: DescribeRecord
	servicecatalog: DescribeServiceAction
	servicecatalog: DescribeServiceActionExecutionParameters
	isableAWSOrganizationsServiceCatalog:D Akses
	servicecatalog: DisassociateBudgetFromResource

Awalan layanan	Tindakan
	servicecatalog: DisassociatePrincipalFromPortfolio
	servicecatalog: DisassociateProductFromPortfolio
	servicecatalog: DisassociateServiceActionFromProvisioningArtifact
	ServiceCatalog:E Akses nableAWSOrganizations
	servicecatalog: ExecuteProvisionedProductPlan
	servicecatalog: ExecuteProvisionedProductServiceAction
	Katalog layanan:G etAWSOrganizations AccessStatus
	servicecatalog: GetProvisionedProductOutputs
	servicecatalog: ImportAsProvisionedProduct
	servicecatalog: ListAcceptedPortfolioShares
	servicecatalog: ListAttributeGroups
	servicecatalog: ListBudgetsForResource
	servicecatalog: ListConstraintsForPortfolio
	servicecatalog: ListLaunchPaths
	servicecatalog: ListOrganizationPortfolioAccess
	servicecatalog: ListPortfolioAccess
	servicecatalog: ListPortfolios
	servicecatalog: ListPortfoliosForProduct
	servicecatalog: ListPrincipalsForPortfolio
	servicecatalog: ListProvisionedProductPlans
	servicecatalog: ListProvisioningArtifacts

Awalan layanan	Tindakan
	servicecatalog: ListProvisioningArtifactsForServiceAction
	servicecatalog: ListRecordHistory
	servicecatalog: ListServiceActions
	servicecatalog: ListServiceActionsForProvisioningArtifact
	servicecatalog: ListStackInstancesForProvisionedProduct
	servicecatalog: NotifyProvisionProductEngineWorkflowResult
	servicecatalog: NotifyTerminateProvisionedProductEngineWorkflowResult
	servicecatalog: NotifyUpdateProvisionedProductEngineWorkflowResult
	servicecatalog: ProvisionProduct
	servicecatalog: RejectPortfolioShare
	servicecatalog: ScanProvisionedProducts
	servicecatalog: SearchProducts
	servicecatalog: SearchProductsAsAdmin
	servicecatalog: SearchProvisionedProducts
	servicecatalog: TerminateProvisionedProduct
	servicecatalog: UpdateConstraint
	servicecatalog: UpdatePortfolio
	servicecatalog: UpdatePortfolioShare
	servicecatalog: UpdateProduct
	servicecatalog: UpdateProvisionedProduct

Awalan layanan	Tindakan
	servicecatalog: UpdateProvisionedProductProperties servicecatalog: UpdateProvisioningArtifact servicecatalog: UpdateServiceAction

Awalan layanan	Tindakan
servicediscovery	penemuan layanan: CreateHttpNamespace penemuan layanan: CreatePrivateDnsNamespace penemuan layanan: CreatePublicDnsNamespace penemuan layanan: CreateService penemuan layanan: DeleteNamespace penemuan layanan: DeleteService penemuan layanan: DeregisterInstance penemuan layanan: GetInstance penemuan layanan: GetInstancesHealthStatus penemuan layanan: GetNamespace penemuan layanan: GetOperation penemuan layanan: GetService penemuan layanan: ListInstances penemuan layanan: ListNamespaces servicediscovery: ListOperations servicediscovery: ListServices servicediscovery: RegisterInstance servicediscovery: UpdateHttpNamespace servicediscovery: UpdateInstanceCustomHealthStatus servicediscovery: UpdatePrivateDnsNamespace servicediscovery: UpdatePublicDnsNamespace

Awalan layanan	Tindakan
	servicediscovery: UpdateService
servicequotas	servicequotas: AssociateServiceQuotaTemplate
	servicequotas: DeleteServiceQuotaIncreaseRequestFromTemplate
	servicequotas: DisassociateServiceQuotaTemplate
	servicequotas: GetAssociationForServiceQuotaTemplate
	ServiceQuotas:G etAWSDefault ServiceQuota
	servicequotas: GetRequestedServiceQuotaChange
	servicequotas: GetServiceQuota
	servicequotas: GetServiceQuotaIncreaseRequestFromTemplate
	ServiceQuotas:L istAWSDefault ServiceQuotas
	servicequotas: ListRequestedServiceQuotaChangeHistory
	servicequotas: ListRequestedServiceQuotaChangeHistoryByQuota
	servicequotas: ListServiceQuotaIncreaseRequestsInTemplate
	servicequotas: ListServiceQuotas
	servicequotas: ListServices
	servicequotas: PutServiceQuotaIncreaseRequestIntoTemplate
	servicequotas: RequestServiceQuotaIncrease

Awalan layanan	Tindakan
ses	ses: BatchGetMetricData ses: CloneReceiptRuleSet ses: CreateAddonInstance ses: CreateAddonSubscription ses: CreateArchive ses: CreateConfigurationSet ses: CreateConfigurationSetEventDestination ses: CreateConfigurationSetTrackingOptions ses: CreateContact ses: CreateContactList ses: CreateCustomVerificationEmailTemplate ses: CreateDedicatedIpPool ses: CreateDeliverabilityTestReport ses: CreateEmailIdentity ses: CreateEmailIdentityPolicy ses: CreateEmailTemplate ses: CreateImportJob ses: CreateIngressPoint ses: CreateReceiptFilter ses: CreateReceiptRule ses: CreateReceiptRuleSet

Awalan layanan	Tindakan
	ses: CreateRelay ses: CreateRuleSet ses: CreateTemplate ses: CreateTrafficPolicy ses: DeleteAddonInstance ses: DeleteAddonSubscription ses: DeleteArchive ses: DeleteConfigurationSet ses: DeleteConfigurationSetEventDestination ses: DeleteConfigurationSetTrackingOptions ses: DeleteContact ses: DeleteContactList ses: DeleteCustomVerificationEmailTemplate ses: DeleteDedicatedIpPool ses: DeleteEmailIdentity ses: DeleteEmailIdentityPolicy ses: DeleteEmailTemplate ses: DeleteIdentity ses: DeleteIdentityPolicy ses: DeleteIngressPoint ses: DeleteReceiptFilter

Awalan layanan	Tindakan
	ses: DeleteReceiptRule ses: DeleteReceiptRuleSet ses: DeleteRelay ses: DeleteRuleSet ses: DeleteSuppressedDestination ses: DeleteTemplate ses: DeleteTrafficPolicy ses: DeleteVerifiedEmailAddress ses: DescribeActiveReceiptRuleSet ses: DescribeConfigurationSet ses: DescribeReceiptRule ses: DescribeReceiptRuleSet ses: GetAccount ses: GetAccountSendingEnabled ses: GetAddonInstance ses: GetAddonSubscription ses: GetArchive ses: GetArchiveExport ses: GetArchiveMessage ses: GetArchiveMessageContent ses: GetArchiveSearch

Awalan layanan	Tindakan
	ses: GetArchiveSearchResults ses: GetBlacklistReports ses: GetConfigurationSet ses: GetConfigurationSetEventDestinations ses: GetContact ses: GetContactList ses: GetCustomVerificationEmailTemplate ses: GetDedicatedIp ses: GetDedicatedIpPool ses: GetDedicatedIps ses: GetDeliverabilityDashboardOptions ses: GetDeliverabilityTestReport ses: GetDomainDeliverabilityCampaign ses: GetDomainStatisticsReport ses: GetEmailIdentity ses: GetEmailIdentityPolicies ses: GetEmailTemplate ses: GetIdentityDkimAttributes ses: GetIdentityMailFromDomainAttributes ses: GetIdentityNotificationAttributes ses: GetIdentityPolicies

Awalan layanan	Tindakan
	ses: GetIdentityVerificationAttributes ses: GetImportJob ses: GetIngressPoint ses: GetMessageInsights ses: GetRelay ses: GetRuleSet ses: GetSendQuota ses: GetSendStatistics ses: GetSuppressedDestination ses: GetTemplate ses: GetTrafficPolicy ses: ListAddonInstances ses: ListAddonSubscriptions ses: ListArchiveExports ses: ListArchives ses: ListArchiveSearches ses: ListConfigurationSets ses: ListContactLists ses: ListContacts ses: ListCustomVerificationEmailTemplates ses: ListDedicatedIpPools

Awalan layanan	Tindakan
	ses: ListDeliverabilityTestReports ses: ListDomainDeliverabilityCampaigns ses: ListEmailIdentities ses: ListEmailTemplates ses: ListExportJobs ses: ListIdentities ses: ListIdentityPolicies ses: ListImportJobs ses: ListIngressPoints ses: ListReceiptFilters ses: ListReceiptRuleSets ses: ListRecommendations ses: ListRelays ses: ListRuleSets ses: ListSuppressedDestinations ses: ListTemplates ses: ListTrafficPolicies ses: ListVerifiedEmailAddresses ses: PutAccountDedicatedIpWarmupAttributes ses: PutAccountDetails ses: PutAccountSendingAttributes

Awalan layanan	Tindakan
	ses: PutAccountSuppressionAttributes ses: PutAccountVdmAttributes ses: PutConfigurationSetDeliveryOptions ses: PutConfigurationSetReputationOptions ses: PutConfigurationSetSendingOptions ses: PutConfigurationSetSuppressionOptions ses: PutConfigurationSetTrackingOptions ses: PutConfigurationSetVdmOptions ses: PutDedicatedIpInPool ses: PutDedicatedIpPoolScalingAttributes ses: PutDedicatedIpWarmupAttributes ses: PutDeliverabilityDashboardOption ses: PutEmailIdentityConfigurationSetAttributes ses: PutEmailIdentityDkimAttributes ses: PutEmailIdentityDkimSigningAttributes ses: PutEmailIdentityFeedbackAttributes ses: PutEmailIdentityMailFromAttributes ses: PutIdentityPolicy ses: PutSuppressedDestination ses: ReorderReceiptRuleSet ses: SendBounce

Awalan layanan	Tindakan
	ses: SendCustomVerificationEmail ses: SetActiveReceiptRuleSet ses: SetIdentityDkimEnabled ses: SetIdentityFeedbackForwardingEnabled ses: SetIdentityHeadersInNotificationsEnabled ses: SetIdentityMailFromDomain ses: SetIdentityNotificationTopic ses: SetReceiptRulePosition ses: StartArchiveExport ses: StartArchiveSearch ses: StopArchiveExport ses: StopArchiveSearch ses: TestRenderEmailTemplate ses: TestRenderTemplate ses: UpdateAccountSendingEnabled ses: UpdateArchive ses: UpdateConfigurationSetEventDestination ses: UpdateConfigurationSetReputationMetricsEnabled ses: UpdateConfigurationSetSendingEnabled ses: UpdateConfigurationSetTrackingOptions ses: UpdateContact

Awalan layanan	Tindakan
	ses: UpdateContactList ses: UpdateCustomVerificationEmailTemplate ses: UpdateEmailIdentityPolicy ses: UpdateEmailTemplate ses: UpdateIngressPoint ses: UpdateReceiptRule ses: UpdateRelay ses: UpdateRuleSet ses: UpdateTemplate ses: UpdateTrafficPolicy ses: VerifyDomainDkim ses: VerifyDomainIdentity ses: VerifyEmailAddress ses: VerifyEmailIdentity

Awalan layanan	Tindakan
melindungi	PerisaissociateDRTLog: Ember
	perisai: AssociateHealthCheck
	perisai: AssociateProactiveEngagementDetails
	perisai: CreateProtection
	perisai: CreateProtectionGroup
	perisai: CreateSubscription
	perisai: DeleteProtection
	perisai: DeleteProtectionGroup
	perisai: DeleteSubscription
	perisai: DescribeAttack
	perisai: DescribeAttackStatistics
	Perisai: D escribeDRTAccess
	perisai: DescribeEmergencyContactSettings
	perisai: DescribeProtection
	perisai: DescribeProtectionGroup
	perisai: DescribeSubscription
	perisai: DisableApplicationLayerAutomaticResponse
	perisai: DisableProactiveEngagement
	PerisaiisassociateDRTLog: D Bucket
	Perisai: D isassociateDRTRole
	perisai: DisassociateHealthCheck

Awalan layanan	Tindakan
	perisai: EnableApplicationLayerAutomaticResponse
	perisai: EnableProactiveEngagement
	perisai: GetSubscriptionState
	perisai: ListAttacks
	perisai: ListProtectionGroups
	perisai: ListProtections
	perisai: ListResourcesInProtectionGroup
	perisai: UpdateApplicationLayerAutomaticResponse
	perisai: UpdateEmergencyContactSettings
	perisai: UpdateProtectionGroup
	perisai: UpdateSubscription

Awalan layanan	Tindakan
penandatangan	penandatangan: AddProfilePermission penandatangan: CancelSigningProfile penandatangan: DescribeSigningJob penandatangan: GetRevocationStatus penandatangan: GetSigningPlatform penandatangan: GetSigningProfile penandatangan: ListProfilePermissions penandatangan: ListSigningJobs penandatangan: ListSigningPlatforms penandatangan: ListSigningProfiles penandatangan: PutSigningProfile penandatangan: RemoveProfilePermission penandatangan: RevokeSignature penandatangan: RevokeSigningProfile penandatangan: SignPayload penandatangan: StartSigningJob

Awalan layanan	Tindakan
simspaceweaver	simspaceweaver: CreateSnapshot
	simspaceweaver: DeleteApp
	simspaceweaver: DeleteSimulation
	simspaceweaver: DescribeApp
	simspaceweaver: DescribeSimulation
	simspaceweaver: ListApps
	simspaceweaver: ListSimulations
	simspaceweaver: StartApp
	simspaceweaver: StartClock
	simspaceweaver: StartSimulation
	simspaceweaver: StopApp
	simspaceweaver: StopClock
	simspaceweaver: StopSimulation

Awalan layanan	Tindakan
sms	sms: CreateApp sms: CreateReplicationJob sms: DeleteApp sms: DeleteAppLaunchConfiguration sms: DeleteAppReplicationConfiguration sms: DeleteAppValidationConfiguration sms: DeleteReplicationJob sms: DeleteServerCatalog sms: DisassociateConnector sms: GenerateChangeSet sms: GenerateTemplate sms: GetApp sms: GetAppLaunchConfiguration sms: GetAppReplicationConfiguration sms: GetAppValidationConfiguration sms: GetAppValidationOutput sms: GetConnectors sms: GetReplicationJobs sms: GetReplicationRuns sms: GetServers sms: ImportAppCatalog

Awalan layanan	Tindakan
	sms: ImportServerCatalog sms: LaunchApp sms: ListApps sms: NotifyAppValidationOutput sms: PutAppLaunchConfiguration sms: PutAppReplicationConfiguration sms: PutAppValidationConfiguration sms: StartAppReplication sms: StartOnDemandAppReplication sms: StartOnDemandReplicationRun sms: StopAppReplication sms: TerminateApp sms: UpdateApp sms: UpdateReplicationJob

Awalan layanan	Tindakan
sms-suara	sms-suara: AssociateProtectConfiguration
	sms-suara: CreateConfigurationSet
	sms-suara: CreateConfigurationSetEventDestination
	sms-suara: CreateEventDestination
	sms-suara: CreateOptOutList
	sms-suara: CreatePool
	sms-suara: CreateProtectConfiguration
	sms-suara: CreateRegistration
	sms-suara: CreateRegistrationAssociation
	sms-suara: CreateRegistrationAttachment
	sms-suara: CreateRegistrationVersion
	sms-suara: CreateVerifiedDestinationNumber
	sms-suara: DeleteAccountDefaultProtectConfiguration
	sms-suara: DeleteConfigurationSet
	sms-suara: DeleteConfigurationSetEventDestination
	sms-suara: DeleteDefaultMessageType
	sms-suara: DeleteDefaultSenderId
	sms-suara: DeleteEventDestination
	sms-suara: DeleteKeyword
	sms-suara: DeleteMediaMessageSpendLimitOverride
	sms-suara: DeleteOptedOutNumber

Awalan layanan	Tindakan
	sms-suara: DeleteOptOutList
	sms-suara: DeletePool
	sms-suara: DeleteProtectConfiguration
	sms-suara: DeleteRegistration
	sms-suara: DeleteRegistrationAttachment
	sms-suara: DeleteTextMessageSpendLimitOverride
	sms-suara: DeleteVerifiedDestinationNumber
	sms-suara: DeleteVoiceMessageSpendLimitOverride
	sms-suara: DescribeAccountAttributes
	sms-suara: DescribeAccountLimits
	sms-suara: DescribeConfigurationSets
	sms-suara: DescribeKeywords
	sms-suara: DescribeOptedOutNumbers
	sms-suara: DescribeOptOutLists
	sms-suara: DescribePhoneNumbers
	sms-suara: DescribePools
	sms-suara: DescribeProtectConfigurations
	sms-suara: DescribeRegistrationAttachments
	sms-suara: DescribeRegistrationFieldDefinitions
	sms-suara: DescribeRegistrationFieldValues
	sms-suara: DescribeRegistrations

Awalan layanan	Tindakan
	sms-suara: DescribeRegistrationSectionDefinitions
	sms-suara: DescribeRegistrationTypeDefinitions
	sms-suara: DescribeRegistrationVersions
	sms-suara: DescribeSenderIds
	sms-suara: DescribeSpendLimits
	sms-suara: DescribeVerifiedDestinationNumbers
	sms-suara: DisassociateOriginationIdentity
	sms-suara: DisassociateProtectConfiguration
	sms-suara: DiscardRegistrationVersion
	sms-suara: GetConfigurationSetEventDestinations
	sms-suara: GetProtectConfigurationCountryRuleSet
	sms-suara: ListConfigurationSets
	sms-suara: ListPoolOriginationIdentities
	sms-suara: ListRegistrationAssociations
	sms-suara: PutKeyword
	sms-suara: PutOptedOutNumber
	sms-suara: ReleasePhoneNumber
	sms-suara: ReleaseSenderId
	sms-suara: RequestPhoneNumber
	sms-suara: RequestSenderId
	sms-suara: SendDestinationNumberVerificationCode

Awalan layanan	Tindakan
	sms-suara: SetAccountDefaultProtectConfiguration
	sms-suara: SetDefaultMessageType
	sms-suara: SetDefaultSenderId
	sms-suara: SetMediaMessageSpendLimitOverride
	sms-suara: SetTextMessageSpendLimitOverride
	sms-suara: SetVoiceMessageSpendLimitOverride
	sms-suara: SubmitRegistrationVersion
	sms-suara: UpdateConfigurationSetEventDestination
	sms-suara: UpdateEventDestination
	sms-suara: UpdatePhoneNumber
	sms-suara: UpdatePool
	sms-suara: UpdateProtectConfiguration
	sms-suara: UpdateProtectConfigurationCountryRuleSet
	sms-suara: UpdateSenderId

Awalan layanan	Tindakan
bola salju	bola salju: CancelCluster bola salju: CancelJob bola salju: CreateAddress bola salju: CreateCluster bola salju: CreateJob bola salju: CreateLongTermPricing bola salju: CreateReturnShippingLabel bola salju: DescribeAddress bola salju: DescribeAddresses bola salju: DescribeCluster bola salju: DescribeJob bola salju: DescribeReturnShippingLabel bola salju: GetJobManifest bola salju: GetJobUnlockCode bola salju: GetSnowballUsage bola salju: GetSoftwareUpdates bola salju: ListClusterJobs bola salju: ListClusters bola salju: ListCompatibleImages bola salju: ListJobs bola salju: ListLongTermPricing

Awalan layanan	Tindakan
	bola salju: ListPickupLocations bola salju: ListServiceVersions bola salju: UpdateCluster bola salju: UpdateJob bola salju: UpdateJobShipmentState bola salju: UpdateLongTermPricing
sqs	persegi: AddPermission persegi: CancelMessageMoveTask persegi: CreateQueue persegi: DeleteQueue persegi: PurgeQueue persegi: RemovePermission persegi: SetQueueAttributes

Awalan layanan	Tindakan
ssm	ssm: AssociateOpsItemRelatedItem ssm: CancelCommand ssm: CancelMaintenanceWindowExecution ssm: CreateActivation ssm: CreateAssociation ssm: CreateAssociationBatch ssm: CreateDocument ssm: CreateMaintenanceWindow ssm: CreateOpsItem ssm: CreateOpsMetadata ssm: CreatePatchBaseline ssm: CreateResourceDataSync ssm: DeleteActivation ssm: DeleteAssociation ssm: DeleteDocument ssm: DeleteInventory ssm: DeleteMaintenanceWindow ssm: DeleteOpsItem ssm: DeleteOpsMetadata ssm: DeleteParameter ssm: DeleteParameters

Awalan layanan	Tindakan
	ssm: DeletePatchBaseline ssm: DeleteResourceDataSync ssm: DeleteResourcePolicy ssm: DeregisterManagedInstance ssm: DeregisterPatchBaselineForPatchGroup ssm: DeregisterTargetFromMaintenanceWindow ssm: DeregisterTaskFromMaintenanceWindow ssm: DescribeActivations ssm: DescribeAssociation ssm: DescribeAssociationExecutions ssm: DescribeAssociationExecutionTargets ssm: DescribeAutomationExecutions ssm: DescribeAutomationStepExecutions ssm: DescribeAvailablePatches ssm: DescribeDocument ssm: DescribeDocumentParameters ssm: DescribeDocumentPermission ssm: DescribeEffectiveInstanceAssociations ssm: DescribeEffectivePatchesForPatchBaseline ssm: DescribeInstanceAssociationsStatus ssm: DescribeInstanceInformation

Awalan layanan	Tindakan
	ssm: DescribeInstancePatches ssm: DescribeInstancePatchStates ssm: DescribeInstancePatchStatesForPatchGroup ssm: DescribeInstanceProperties ssm: DescribeInventoryDeletions ssm: DescribeMaintenanceWindowExecutions ssm: DescribeMaintenanceWindowExecutionTaskInvocations ssm: DescribeMaintenanceWindowExecutionTasks ssm: DescribeMaintenanceWindows ssm: DescribeMaintenanceWindowSchedule ssm: DescribeMaintenanceWindowsForTarget ssm: DescribeMaintenanceWindowTargets ssm: DescribeMaintenanceWindowTasks ssm: DescribeOpsItems ssm: DescribeParameters ssm: DescribePatchBaselines ssm: DescribePatchGroups ssm: DescribePatchGroupState ssm: DescribePatchProperties ssm: DescribeSessions ssm: DisassociateOpsItemRelatedItem

Awalan layanan	Tindakan
	ssm: GetAutomationExecution ssm: GetCalendarState ssm: GetCommandInvocation ssm: GetConnectionStatus ssm: GetDefaultPatchBaseline ssm: GetDeployablePatchSnapshotForInstance ssm: GetDocument ssm: GetInventory ssm: GetInventorySchema ssm: GetMaintenanceWindow ssm: GetMaintenanceWindowExecution ssm: GetMaintenanceWindowExecutionTask ssm: GetMaintenanceWindowExecutionTaskInvocation ssm: GetMaintenanceWindowTask ssm: GetOpsItem ssm: GetOpsMetadata ssm: GetOpsSummary ssm: GetParameter ssm: GetParameterHistory ssm: GetParameters ssm: GetParametersByPath

Awalan layanan	Tindakan
	ssm: GetPatchBaseline ssm: GetPatchBaselineForPatchGroup ssm: GetResourcePolicies ssm: GetServiceSetting ssm: LabelParameterVersion ssm: ListAssociations ssm: ListAssociationVersions ssm: ListCommandInvocations ssm: ListCommands ssm: ListComplianceItems ssm: ListComplianceSummaries ssm: ListDocumentMetadataHistory ssm: ListDocuments ssm: ListDocumentVersions ssm: ListInstanceAssociations ssm: ListInventoryEntries ssm: ListOpsItemEvents ssm: ListOpsItemRelatedItems ssm: ListOpsMetadata ssm: ListResourceComplianceSummaries ssm: ListResourceDataSync

Awalan layanan	Tindakan
	ssm: ModifyDocumentPermission ssm: PutComplianceItems ssm: PutInventory ssm: PutParameter ssm: PutResourcePolicy ssm: RegisterDefaultPatchBaseline ssm: RegisterManagedInstance ssm: RegisterPatchBaselineForPatchGroup ssm: RegisterTargetWithMaintenanceWindow ssm: RegisterTaskWithMaintenanceWindow ssm: ResetServiceSetting ssm: ResumeSession ssm: SendAutomationSignal ssm: SendCommand ssm: StartAssociationsOnce ssm: StartAutomationExecution ssm: StartChangeRequestExecution ssm: StartSession ssm: StopAutomationExecution ssm: TerminateSession ssm: UnlabelParameterVersion

Awalan layanan	Tindakan
	ssm: UpdateAssociation ssm: UpdateAssociationStatus ssm: UpdateDocument ssm: UpdateDocumentDefaultVersion ssm: UpdateDocumentMetadata ssm: UpdateInstanceInformation ssm: UpdateMaintenanceWindow ssm: UpdateMaintenanceWindowTarget ssm: UpdateMaintenanceWindowTask ssm: UpdateManagedInstanceRole ssm: UpdateOpsItem ssm: UpdateOpsMetadata ssm: UpdatePatchBaseline ssm: UpdateResourceDataSync ssm: UpdateServiceSetting

Awalan layanan	Tindakan
insiden ssm	insiden ssm-: BatchGetIncidentFindings
	insiden ssm-: CreateReplicationSet
	insiden ssm-: CreateResponsePlan
	insiden ssm-: CreateTimelineEvent
	insiden ssm-: DeleteIncidentRecord
	insiden ssm-: DeleteReplicationSet
	insiden ssm-: DeleteResourcePolicy
	insiden ssm-: DeleteResponsePlan
	insiden ssm-: DeleteTimelineEvent
	insiden ssm-: GetIncidentRecord
	insiden ssm-: GetReplicationSet
	insiden ssm-: GetResourcePolicies
	insiden ssm-: GetResponsePlan
	insiden ssm-: GetTimelineEvent
	insiden ssm-: ListIncidentFindings
	insiden ssm-: ListIncidentRecords
	insiden ssm-: ListRelatedItems
	insiden ssm-: ListReplicationSets
	insiden ssm-: ListResponsePlans
	insiden ssm-: ListTimelineEvents
	insiden ssm-: PutResourcePolicy

Awalan layanan	Tindakan
	insiden ssm-: StartIncident
	insiden ssm-: UpdateDeletionProtection
	insiden ssm-: UpdateIncidentRecord
	insiden ssm-: UpdateRelatedItems
	insiden ssm-: UpdateReplicationSet
	insiden ssm-: UpdateResponsePlan
	insiden ssm-: UpdateTimelineEvent

Awalan layanan	Tindakan
ssm-sap	ssm-sap: BackupDatabase ssm-sap: DeleteResourcePermission ssm-sap: DeregisterApplication ssm-sap: GetApplication ssm-sap: GetComponent ssm-sap: GetDatabase ssm-sap: GetOperation ssm-sap: GetResourcePermission ssm-sap: ListApplications ssm-sap: ListComponents ssm-sap: ListDatabases ssm-sap: ListOperationEvents ssm-sap: ListOperations ssm-sap: PutResourcePermission ssm-sap: RegisterApplication ssm-sap: RestoreDatabase ssm-sap: StartApplication ssm-sap: StartApplicationRefresh ssm-sap: StopApplication ssm-sap: UpdateApplicationSettings SSM-SAPpdateHANABackup: Pengaturan U

Awalan layanan	Tindakan
negara	negara bagian: CreateActivity
	negara bagian: CreateStateMachine
	negara bagian: CreateStateMachineAlias
	negara bagian: DeleteActivity
	negara bagian: DeleteStateMachine
	negara bagian: DeleteStateMachineAlias
	negara bagian: DeleteStateMachineVersion
	negara bagian: DescribeActivity
	negara bagian: DescribeExecution
	negara bagian: DescribeMapRun
	negara bagian: DescribeStateMachine
	negara bagian: DescribeStateMachineAlias
	negara bagian: DescribeStateMachineForExecution
	negara bagian: GetExecutionHistory
	negara bagian: ListActivities
	negara bagian: ListExecutions
	negara bagian: ListMapRuns
	negara bagian: ListStateMachineAliases
	negara bagian: ListStateMachines
	negara bagian: ListStateMachineVersions
	negara bagian: SendTaskFailure

Awalan layanan	Tindakan
	negara bagian: SendTaskHeartbeat negara bagian: SendTaskSuccess negara bagian: StartExecution negara bagian: StopExecution negara bagian: UpdateMapRun negara bagian: UpdateStateMachine negara bagian: UpdateStateMachineAlias negara bagian: ValidateStateMachineDefinition
sts	sts: AssumeRole sts: AssumeRoleWith SAML sts: AssumeRoleWithWebIdentity sts: DecodeAuthorizationMessage sts: GetAccessKeyInfo sts: GetCallerIdentity sts: GetFederationToken sts: GetSessionToken

Awalan layanan	Tindakan
swf	swf: DeleteActivityType
	swf: DeleteWorkflowType
	swf: DeprecateActivityType
	swf: DeprecateDomain
	swf: DeprecateWorkflowType
	swf: DescribeActivityType
	swf: DescribeDomain
	swf: DescribeWorkflowType
	swf: ListActivityTypes
	swf: ListDomains
	swf: ListWorkflowTypes
	swf: RegisterActivityType
	swf: RegisterDomain
	swf: RegisterWorkflowType
	swf: UndeprecateActivityType
	swf: UndeprecateDomain
	swf: UndeprecateWorkflowType

Awalan layanan	Tindakan
sintetis	sintetis: AssociateResource sintetis: CreateCanary sintetis: CreateGroup sintetis: DeleteCanary sintetis: DeleteGroup sintetis: DescribeCanaries sintetis: DescribeCanariesLastRun sintetis: DescribeRuntimeVersions sintetis: DisassociateResource sintetis: GetCanary sintetis: GetCanaryRuns sintetis: GetGroup sintetis: ListAssociatedGroups sintetis: ListGroupResources sintetis: ListGroups sintetis: StartCanary sintetis: StopCanary sintetis: UpdateCanary

Awalan layanan	Tindakan
tanda	Tag: DescribeReportCreation Tag: GetComplianceSummary Tag: GetResources Tag: StartReportCreation

Awalan layanan	Tindakan
textract	teks: AnalyzeDocument teks: AnalyzeExpense Textract:AnalyzeID teks: CreateAdapter teks: CreateAdapterVersion teks: DeleteAdapter teks: DeleteAdapterVersion teks: DetectDocumentText teks: GetAdapter teks: GetAdapterVersion teks: GetDocumentAnalysis teks: GetDocumentTextDetection teks: GetExpenseAnalysis teks: GetLendingAnalysis teks: GetLendingAnalysisSummary teks: ListAdapters teks: ListAdapterVersions teks: StartDocumentAnalysis teks: StartDocumentTextDetection teks: StartExpenseAnalysis teks: StartLendingAnalysis

Awalan layanan	Tindakan
	teks: UpdateAdapter

Awalan layanan	Tindakan
aliran waktu	aliran waktu: CancelQuery
	aliran waktu: CreateDatabase
	aliran waktu: CreateScheduledQuery
	aliran waktu: CreateTable
	aliran waktu: DeleteDatabase
	aliran waktu: DeleteScheduledQuery
	aliran waktu: DeleteTable
	aliran waktu: DescribeAccountSettings
	aliran waktu: DescribeDatabase
	aliran waktu: DescribeScheduledQuery
	aliran waktu: DescribeTable
	aliran waktu: ExecuteScheduledQuery
	aliran waktu: ListBatchLoadTasks
	aliran waktu: ListDatabases
	aliran waktu: ListScheduledQueries
	aliran waktu: ListTables
	aliran waktu: PrepareQuery
	aliran waktu: UpdateAccountSettings
	aliran waktu: UpdateDatabase
	aliran waktu: UpdateScheduledQuery
	aliran waktu: UpdateTable

Awalan layanan	Tindakan
tnb	tnb: CancelSolNetworkOperation tnb: CreateSolFunctionPackage tnb: CreateSolNetworkInstance tnb: CreateSolNetworkPackage tnb: DeleteSolFunctionPackage tnb: DeleteSolNetworkInstance tnb: DeleteSolNetworkPackage tnb: GetSolFunctionInstance tnb: GetSolFunctionPackage tnb: GetSolFunctionPackageContent tnb: GetSolFunctionPackageDescriptor tnb: GetSolNetworkInstance tnb: GetSolNetworkOperation tnb: GetSolNetworkPackage tnb: GetSolNetworkPackageContent tnb: GetSolNetworkPackageDescriptor tnb: InstantiateSolNetworkInstance tnb: ListSolFunctionInstances tnb: ListSolFunctionPackages tnb: ListSolNetworkInstances tnb: ListSolNetworkOperations

Awalan layanan	Tindakan
	tnb: ListSolNetworkPackages tnb: PutSolFunctionPackageContent tnb: PutSolNetworkPackageContent tnb: TerminateSolNetworkInstance tnb: UpdateSolFunctionPackage tnb: UpdateSolNetworkInstance tnb: UpdateSolNetworkPackage tnb: ValidateSolFunctionPackageContent tnb: ValidateSolNetworkPackageContent

Awalan layanan	Tindakan
mentranskripsikan	transkripsikan: CreateCallAnalyticsCategory
	transkripsikan: CreateLanguageModel
	transkripsikan: CreateMedicalVocabulary
	transkripsikan: CreateVocabulary
	transkripsikan: CreateVocabularyFilter
	transkripsikan: DeleteCallAnalyticsCategory
	transkripsikan: DeleteCallAnalyticsJob
	transkripsikan: DeleteLanguageModel
	transkripsikan: DeleteMedicalScribeJob
	transkripsikan: DeleteMedicalTranscriptionJob
	transkripsikan: DeleteMedicalVocabulary
	transkripsikan: DeleteTranscriptionJob
	transkripsikan: DeleteVocabulary
	transkripsikan: DeleteVocabularyFilter
	transkripsikan: DescribeLanguageModel
	transkripsikan: GetCallAnalyticsCategory
	transkripsikan: GetCallAnalyticsJob
	transkripsikan: GetMedicalScribeJob
	transkripsikan: GetMedicalTranscriptionJob
	transkripsikan: GetMedicalVocabulary
	transkripsikan: GetTranscriptionJob

Awalan layanan	Tindakan
	transkripsikan: GetVocabulary
	transkripsikan: GetVocabularyFilter
	transkripsikan: ListCallAnalyticsCategories
	transkripsikan: ListCallAnalyticsJobs
	transkripsikan: ListLanguageModels
	transkripsikan: ListMedicalScribeJobs
	transkripsikan: ListMedicalTranscriptionJobs
	transkripsikan: ListMedicalVocabularies
	transkripsikan: ListTranscriptionJobs
	transkripsikan: ListVocabularies
	transkripsikan: ListVocabularyFilters
	transkripsikan: StartCallAnalyticsJob
	transkripsikan: StartCallAnalyticsStreamTranscription
	transkripsikan: StartCallAnalyticsStreamTranscriptionWebSocket
	transkripsikan: StartMedicalScribeJob
	transkripsikan: StartMedicalStreamTranscription
	transkripsikan: StartMedicalStreamTranscriptionWebSocket
	transkripsikan: StartMedicalTranscriptionJob
	transkripsikan: StartStreamTranscription
	transkripsikan: StartStreamTranscriptionWebSocket
	transkripsikan: StartTranscriptionJob

Awalan layanan	Tindakan
	transkripsikan: UpdateCallAnalyticsCategory transkripsikan: UpdateMedicalVocabulary transkripsikan: UpdateVocabulary transkripsikan: UpdateVocabularyFilter

Awalan layanan	Tindakan
transfer	transfer: CreateAccess transfer: CreateAgreement transfer: CreateConnector transfer: CreateProfile transfer: CreateServer transfer: CreateUser transfer: CreateWorkflow transfer: DeleteAccess transfer: DeleteAgreement transfer: DeleteCertificate transfer: DeleteConnector transfer: DeleteHostKey transfer: DeleteProfile transfer: DeleteServer transfer: DeleteSshPublicKey transfer: DeleteUser transfer: DeleteWorkflow transfer: DescribeAccess transfer: DescribeAgreement transfer: DescribeCertificate transfer: DescribeConnector

Awalan layanan	Tindakan
	transfer: DescribeExecution transfer: DescribeHostKey transfer: DescribeProfile transfer: DescribeSecurityPolicy transfer: DescribeServer transfer: DescribeUser transfer: DescribeWorkflow transfer: ImportCertificate transfer: ImportHostKey transfer: ImportSshPublicKey transfer: ListAccesses transfer: ListCertificates transfer: ListConnectors transfer: ListExecutions transfer: ListHostKeys transfer: ListProfiles transfer: ListSecurityPolicies transfer: ListServers transfer: ListUsers transfer: ListWorkflows transfer: SendWorkflowStepState

Awalan layanan	Tindakan
	transfer: StartDirectoryListing transfer: StartFileTransfer transfer: StartServer transfer: StopServer transfer: TestConnection transfer: TestIdentityProvider transfer: UpdateAccess transfer: UpdateAgreement transfer: UpdateCertificate transfer: UpdateConnector transfer: UpdateHostKey transfer: UpdateProfile transfer: UpdateServer transfer: UpdateUser

Awalan layanan	Tindakan
menerjemahkan	menerjemahkan: CreateParallelData
	menerjemahkan: DeleteParallelData
	menerjemahkan: DeleteTerminology
	menerjemahkan: DescribeTextTranslationJob
	menerjemahkan: GetParallelData
	menerjemahkan: GetTerminology
	menerjemahkan: ImportTerminology
	menerjemahkan: ListLanguages
	menerjemahkan: ListParallelData
	menerjemahkan: ListTerminologies
	menerjemahkan: ListTextTranslationJobs
	menerjemahkan: StartTextTranslationJob
	menerjemahkan: StopTextTranslationJob
	menerjemahkan: TranslateDocument
	menerjemahkan: TranslateText
	menerjemahkan: UpdateParallelData

Awalan layanan	Tindakan
voiceid	voiceid: AssociateFraudster voiceid: CreateDomain voiceid: CreateWatchlist voiceid: DeleteDomain voiceid: DeleteFraudster voiceid: DeleteSpeaker voiceid: DeleteWatchlist voiceid: DescribeDomain voiceid: DescribeFraudster voiceid: DescribeFraudsterRegistrationJob voiceid: DescribeSpeaker voiceid: DescribeSpeakerEnrollmentJob voiceid: DescribeWatchlist voiceid: DisassociateFraudster voiceid: EvaluateSession voiceid: ListDomains voiceid: ListFraudsterRegistrationJobs voiceid: ListFraudsters voiceid: ListSpeakerEnrollmentJobs voiceid: ListSpeakers voiceid: ListWatchlists

Awalan layanan	Tindakan
	voiceid: OptOutSpeaker voiceid: StartFraudsterRegistrationJob voiceid: StartSpeakerEnrollmentJob voiceid: UpdateDomain voiceid: UpdateWatchlist

Awalan layanan	Tindakan
vpc-kisi	vpc-kisi: CreateAccessLogSubscription
	vpc-kisi: CreateListener
	vpc-kisi: CreateRule
	vpc-kisi: CreateService
	vpc-kisi: CreateServiceNetwork
	vpc-kisi: CreateServiceNetworkServiceAssociation
	vpc-kisi: CreateServiceNetworkVpcAssociation
	vpc-kisi: CreateTargetGroup
	vpc-kisi: DeleteAccessLogSubscription
	vpc-kisi: DeleteAuthPolicy
	vpc-kisi: DeleteListener
	vpc-kisi: DeleteResourcePolicy
	vpc-kisi: DeleteRule
	vpc-kisi: DeleteService
	vpc-kisi: DeleteServiceNetwork
	vpc-kisi: DeleteServiceNetworkServiceAssociation
	vpc-kisi: DeleteServiceNetworkVpcAssociation
	vpc-kisi: DeleteTargetGroup
	vpc-kisi: DeregisterTargets
	vpc-kisi: GetAccessLogSubscription
	vpc-kisi: GetAuthPolicy

Awalan layanan	Tindakan
	vpc-kisi: GetListener
	vpc-kisi: GetResourcePolicy
	vpc-kisi: GetRule
	vpc-kisi: GetService
	vpc-kisi: GetServiceNetwork
	vpc-kisi: GetServiceNetworkServiceAssociation
	vpc-kisi: GetServiceNetworkVpcAssociation
	vpc-kisi: GetTargetGroup
	vpc-kisi: ListAccessLogSubscriptions
	vpc-kisi: ListListeners
	vpc-kisi: ListRules
	vpc-kisi: ListServiceNetworks
	vpc-kisi: ListServiceNetworkServiceAssociations
	vpc-kisi: ListServiceNetworkVpcAssociations
	vpc-kisi: ListServices
	vpc-kisi: ListTargetGroups
	vpc-kisi: ListTargets
	vpc-kisi: PutAuthPolicy
	vpc-kisi: PutResourcePolicy
	vpc-kisi: RegisterTargets
	vpc-kisi: UpdateAccessLogSubscription

Awalan layanan	Tindakan
	vpc-kisi: UpdateListener vpc-kisi: UpdateRule vpc-kisi: UpdateService vpc-kisi: UpdateServiceNetwork vpc-kisi: UpdateServiceNetworkVpcAssociation vpc-kisi: UpdateTargetGroup

Awalan layanan	Tindakan
wafv2	wafv2: AssociateWeb ACL wafv2: CheckCapacity WAFv2: C reateAPIKey WAFv2: C reateIPSet wafv2: CreateRegexPatternSet wafv2: CreateRuleGroup wafv2: CreateWeb ACL Wafv2:D eleteAPIKey wafv2: DeleteFirewallManagerRuleGroups Wafv2:D eleteIPSet wafv2: DeleteLoggingConfiguration wafv2: DeletePermissionPolicy wafv2: DeleteRegexPatternSet wafv2: DeleteRuleGroup wafv2: DeleteWeb ACL wafv2: DescribeAllManagedProducts wafv2: DescribeManagedProductsByVendor wafv2: DescribeManagedRuleGroup wafv2: DisassociateWeb ACL wafv2: GenerateMobileSdkReleaseUrl wafv2: GetDecrypted APIKey

Awalan layanan	Tindakan
	WAFv2: g etIPSet wafv2: GetLoggingConfiguration wafv2: GetManagedRuleSet wafv2: GetMobileSdkRelease wafv2: GetPermissionPolicy wafv2: GetRateBasedStatementManagedKeys wafv2: GetRegexPatternSet wafv2: GetRuleGroup wafv2: GetSampledRequests wafv2: GetWeb ACLForResource Wafv2: L istAPIKeys wafv2: ListAvailableManagedRuleGroups wafv2: ListAvailableManagedRuleGroupVersions Wafv2: L istIPSets wafv2: ListLoggingConfigurations wafv2: ListManagedRuleSets wafv2: ListMobileSdkReleases wafv2: ListRegexPatternSets wafv2: ListResourcesForWeb ACL wafv2: ListRuleGroups wafv2: ListWeb ACLs

Awalan layanan	Tindakan
	wafv2: PutLoggingConfiguration wafv2: PutManagedRuleSetVersions wafv2: PutPermissionPolicy Wafv2:u pdateIPSet wafv2: UpdateManagedRuleSetVersionExpiryDate wafv2: UpdateRegexPatternSet wafv2: UpdateRuleGroup wafv2: UpdateWeb ACL

Awalan layanan	Tindakan
wellarchitected	wellarchitected: AssociateLenses
	wellarchitected: AssociateProfiles
	wellarchitected: CreateLensShare
	wellarchitected: CreateLensVersion
	wellarchitected: CreateMilestone
	wellarchitected: CreateProfile
	wellarchitected: CreateProfileShare
	wellarchitected: CreateReviewTemplate
	wellarchitected: CreateWorkload
	wellarchitected: CreateWorkloadShare
	wellarchitected: DeleteLens
	wellarchitected: DeleteLensShare
	wellarchitected: DeleteProfile
	wellarchitected: DeleteProfileShare
	wellarchitected: DeleteReviewTemplate
	wellarchitected: DeleteTemplateShare
	wellarchitected: DeleteWorkload
	wellarchitected: DeleteWorkloadShare
	wellarchitected: DisassociateLenses
	wellarchitected: DisassociateProfiles
	wellarchitected: ExportLens

Awalan layanan	Tindakan
	wellarchitected: GetAnswer
	wellarchitected: GetConsolidatedReport
	wellarchitected: GetGlobalSettings
	wellarchitected: GetLens
	wellarchitected: GetLensReview
	wellarchitected: GetLensReviewReport
	wellarchitected: GetLensVersionDifference
	wellarchitected: GetMilestone
	wellarchitected: GetProfile
	wellarchitected: GetProfileTemplate
	wellarchitected: GetReviewTemplate
	wellarchitected: GetReviewTemplateAnswer
	wellarchitected: GetReviewTemplateLensReview
	wellarchitected: GetWorkload
	wellarchitected: ImportLens
	wellarchitected: ListAnswers
	wellarchitected: ListCheckDetails
	wellarchitected: ListCheckSummaries
	wellarchitected: ListLenses
	wellarchitected: ListLensReviewImprovements
	wellarchitected: ListLensReviews

Awalan layanan	Tindakan
	wellarchitected: ListLensShares
	wellarchitected: ListMilestones
	wellarchitected: ListNotifications
	wellarchitected: ListProfileNotifications
	wellarchitected: ListProfiles
	wellarchitected: ListProfileShares
	wellarchitected: ListReviewTemplateAnswers
	wellarchitected: ListReviewTemplates
	wellarchitected: ListShareInvitations
	wellarchitected: ListTemplateShares
	wellarchitected: ListWorkloads
	wellarchitected: ListWorkloadShares
	wellarchitected: UpdateAnswer
	wellarchitected: UpdateGlobalSettings
	wellarchitected: UpdateIntegration
	wellarchitected: UpdateLensReview
	wellarchitected: UpdateProfile
	wellarchitected: UpdateReviewTemplate
	wellarchitected: UpdateReviewTemplateLensReview
	wellarchitected: UpdateShareInvitation
	wellarchitected: UpdateWorkload

Awalan layanan	Tindakan
	wellarchitected: UpdateWorkloadShare wellarchitected: UpgradeLensReview wellarchitected: UpgradeProfileVersion wellarchitected: UpgradeReviewTemplateLensReview

Awalan layanan	Tindakan
kebijaksanaan	kebijaksanaan: CreateAssistant kebijaksanaan: CreateAssistantAssociation kebijaksanaan: CreateContent kebijaksanaan: CreateKnowledgeBase kebijaksanaan: CreateQuickResponse kebijaksanaan: CreateSession kebijaksanaan: DeleteAssistant kebijaksanaan: DeleteAssistantAssociation kebijaksanaan: DeleteContent kebijaksanaan: DeleteImportJob kebijaksanaan: DeleteKnowledgeBase kebijaksanaan: DeleteQuickResponse kebijaksanaan: GetAssistant kebijaksanaan: GetAssistantAssociation kebijaksanaan: GetContent kebijaksanaan: GetContentAssociation kebijaksanaan: GetContentSummary kebijaksanaan: GetImportJob kebijaksanaan: GetKnowledgeBase kebijaksanaan: GetRecommendations kebijaksanaan: GetSession

Awalan layanan	Tindakan
	kebijaksanaan: ListAssistantAssociations kebijaksanaan: ListAssistants kebijaksanaan: ListContentAssociations kebijaksanaan: ListContents kebijaksanaan: ListImportJobs kebijaksanaan: ListKnowledgeBases kebijaksanaan: ListQuickResponses kebijaksanaan: NotifyRecommendationsReceived kebijaksanaan: QueryAssistant kebijaksanaan: RemoveKnowledgeBaseTemplateUri kebijaksanaan: SearchContent kebijaksanaan: SearchQuickResponses kebijaksanaan: SearchSessions kebijaksanaan: StartContentUpload kebijaksanaan: StartImportJob kebijaksanaan: UpdateContent kebijaksanaan: UpdateKnowledgeBaseTemplateUri kebijaksanaan: UpdateQuickResponse kebijaksanaan: UpdateSession

Awalan layanan	Tindakan
tautan kerja	tautan kerja: AssociateDomain
	tautan kerja: AssociateWebsiteAuthorizationProvider
	tautan kerja: AssociateWebsiteCertificateAuthority
	tautan kerja: CreateFleet
	tautan kerja: DeleteFleet
	tautan kerja: DescribeAuditStreamConfiguration
	tautan kerja: DescribeCompanyNetworkConfiguration
	tautan kerja: DescribeDevice
	tautan kerja: DescribeDevicePolicyConfiguration
	tautan kerja: DescribeDomain
	tautan kerja: DescribeFleetMetadata
	tautan kerja: DescribeIdentityProviderConfiguration
	tautan kerja: DescribeWebsiteCertificateAuthority
	tautan kerja: DisassociateDomain
	tautan kerja: DisassociateWebsiteAuthorizationProvider
	tautan kerja: DisassociateWebsiteCertificateAuthority
	tautan kerja: ListDevices
	tautan kerja: ListDomains
	tautan kerja: ListFleets
	tautan kerja: ListWebsiteAuthorizationProviders
	tautan kerja: ListWebsiteCertificateAuthorities

Awalan layanan	Tindakan
	tautan kerja: RestoreDomainAccess
	tautan kerja: RevokeDomainAccess
	tautan kerja: SignOutUser
	tautan kerja: UpdateAuditStreamConfiguration
	tautan kerja: UpdateCompanyNetworkConfiguration
	tautan kerja: UpdateDevicePolicyConfiguration
	tautan kerja: UpdateDomainMetadata
	tautan kerja: UpdateFleetMetadata
	tautan kerja: UpdateIdentityProviderConfiguration

Awalan layanan	Tindakan
ruang kerja	ruang kerja: AcceptAccountLinkInvitation ruang kerja: AssociateConnectionAlias ruang kerja: AssociateIpGroups ruang kerja: AssociateWorkspaceApplication ruang kerja: CopyWorkspaceImage ruang kerja: CreateAccountLinkInvitation ruang kerja: CreateConnectClientAddIn ruang kerja: CreateConnectionAlias ruang kerja: CreateIpGroup ruang kerja: CreateStandbyWorkspaces ruang kerja: CreateUpdatedWorkspaceImage ruang kerja: CreateWorkspaceBundle ruang kerja: CreateWorkspaceImage ruang kerja: CreateWorkspaces ruang kerja: CreateWorkspacesPool ruang kerja: DeleteAccountLinkInvitation ruang kerja: DeleteClientBranding ruang kerja: DeleteConnectClientAddIn ruang kerja: DeleteConnectionAlias ruang kerja: DeleteIpGroup ruang kerja: DeleteWorkspaceBundle

Awalan layanan	Tindakan
	ruang kerja: DeleteWorkspaceImage
	ruang kerja: DeployWorkspaceApplications
	ruang kerja: DeregisterWorkspaceDirectory
	ruang kerja: DescribeAccount
	ruang kerja: DescribeAccountModifications
	ruang kerja: DescribeApplicationAssociations
	ruang kerja: DescribeApplications
	ruang kerja: DescribeBundleAssociations
	ruang kerja: DescribeClientBranding
	ruang kerja: DescribeClientProperties
	ruang kerja: DescribeConnectClientAddIns
	ruang kerja: DescribeConnectionAliases
	ruang kerja: DescribeConnectionAliasPermissions
	ruang kerja: DescribeImageAssociations
	ruang kerja: DescribeIpGroups
	ruang kerja: DescribeWorkspaceAssociations
	ruang kerja: DescribeWorkspaceBundles
	ruang kerja: DescribeWorkspaceDirectories
	ruang kerja: DescribeWorkspaceImagePermissions
	ruang kerja: DescribeWorkspaces
	ruang kerja: DescribeWorkspacesConnectionStatus

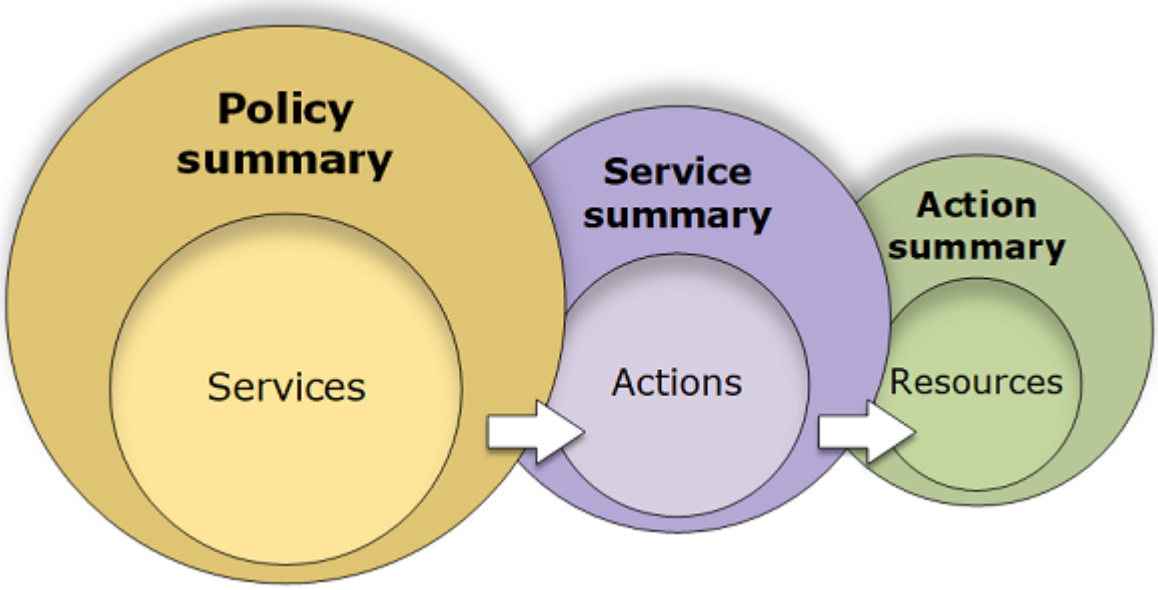
Awalan layanan	Tindakan
	ruang kerja: DescribeWorkspaceSnapshots ruang kerja: DescribeWorkspacesPools ruang kerja: DescribeWorkspacesPoolSessions ruang kerja: DisassociateConnectionAlias ruang kerja: DisassociateIpGroups ruang kerja: DisassociateWorkspaceApplication ruang kerja: GetAccountLink ruang kerja: ImportClientBranding ruang kerja: ImportWorkspaceImage ruang kerja: ListAccountLinks ruang kerja: ListAvailableManagementCidrRanges ruang kerja: MigrateWorkspace ruang kerja: ModifyAccount ruang kerja: ModifyCertificateBasedAuthProperties ruang kerja: ModifyClientProperties ruang kerja: ModifySamlProperties ruang kerja: ModifySelfservicePermissions ruang kerja: ModifyStreamingProperties ruang kerja: ModifyWorkspaceAccessProperties ruang kerja: ModifyWorkspaceCreationProperties ruang kerja: ModifyWorkspaceProperties

Awalan layanan	Tindakan
	ruang kerja: ModifyWorkspaceState ruang kerja: RebootWorkspaces ruang kerja: RebuildWorkspaces ruang kerja: RegisterWorkspaceDirectory ruang kerja: RejectAccountLinkInvitation ruang kerja: RestoreWorkspace ruang kerja: StartWorkspaces ruang kerja: StartWorkspacesPool ruang kerja: StopWorkspaces ruang kerja: StopWorkspacesPool ruang kerja: TerminateWorkspaces ruang kerja: TerminateWorkspacesPool ruang kerja: TerminateWorkspacesPoolSession ruang kerja: UpdateConnectClientAddIn ruang kerja: UpdateConnectionAliasPermission ruang kerja: UpdateWorkspaceBundle ruang kerja: UpdateWorkspaceImagePermission ruang kerja: UpdateWorkspacesPool

Awalan layanan	Tindakan
xray	xray: CreateGroup
	xray: CreateSamplingRule
	xray: DeleteGroup
	xray: DeleteResourcePolicy
	xray: DeleteSamplingRule
	xray: GetEncryptionConfig
	xray: GetGroup
	xray: GetGroups
	xray: GetInsight
	xray: GetInsightEvents
	xray: GetInsightImpactGraph
	xray: GetInsightSummaries
	xray: GetSamplingRules
	xray: ListResourcePolicies
	xray: PutEncryptionConfig
	xray: PutResourcePolicy
	xray: UpdateGroup
	xray: UpdateSamplingRule

Ringkasan kebijakan

IAMKonsol menyertakan tabel ringkasan kebijakan yang menjelaskan tingkat akses, sumber daya, dan kondisi yang diizinkan atau ditolak untuk setiap layanan dalam kebijakan. Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan kebijakan mencakup daftar layanan. Pilih layanan yang tertera untuk melihat ringkasan layanan. Tabel ringkasan ini mencakup daftar tindakan dan izin terkait untuk layanan yang dipilih. Anda dapat memilih tindakan dari tabel tersebut untuk melihat ringkasan tindakan. Tabel ini mencakup daftar sumber daya dan IT untuk tindakan yang dipilih.



Anda dapat melihat rangkuman kebijakan di halaman Pengguna atau Peran untuk semua kebijakan (terkelola dan inline) yang terlampir pada pengguna tersebut. Lihat rangkuman dalam halaman Kebijakan untuk seluruh kebijakan terkelola Kebijakan AWS terkelola mencakup kebijakan AWS terkelola, kebijakan fungsi pekerjaan terkelola, dan kebijakan yang dikelola pelanggan. Anda dapat melihat ringkasan untuk kebijakan ini di halaman Kebijakan terlepas dari apakah mereka dilampirkan ke pengguna atau identitas lainnyaIAM.

Anda dapat menggunakan informasi dalam rangkuman kebijakan untuk memahami izin yang diizinkan atau ditolak oleh kebijakan Anda. Ringkasan kebijakan dapat membantu Anda pemecahan masalah dan memperbaiki kebijakan yang tidak memberikan izin yang Anda harapkan.

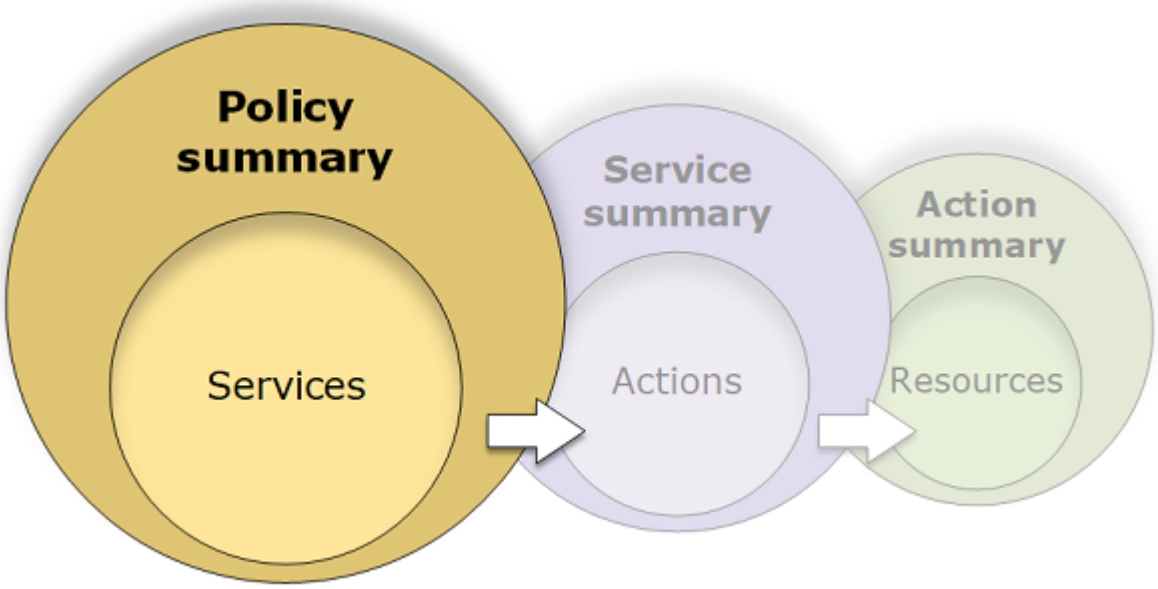
Topik

- Ringkasan kebijakan (daftar layanan)
- Tingkat akses dalam ringkasan kebijakan

- Ringkasan layanan (daftar tindakan)
- Ringkasan tindakan (daftar sumber daya)
- Contoh ringkasan kebijakan

Ringkasan kebijakan (daftar layanan)

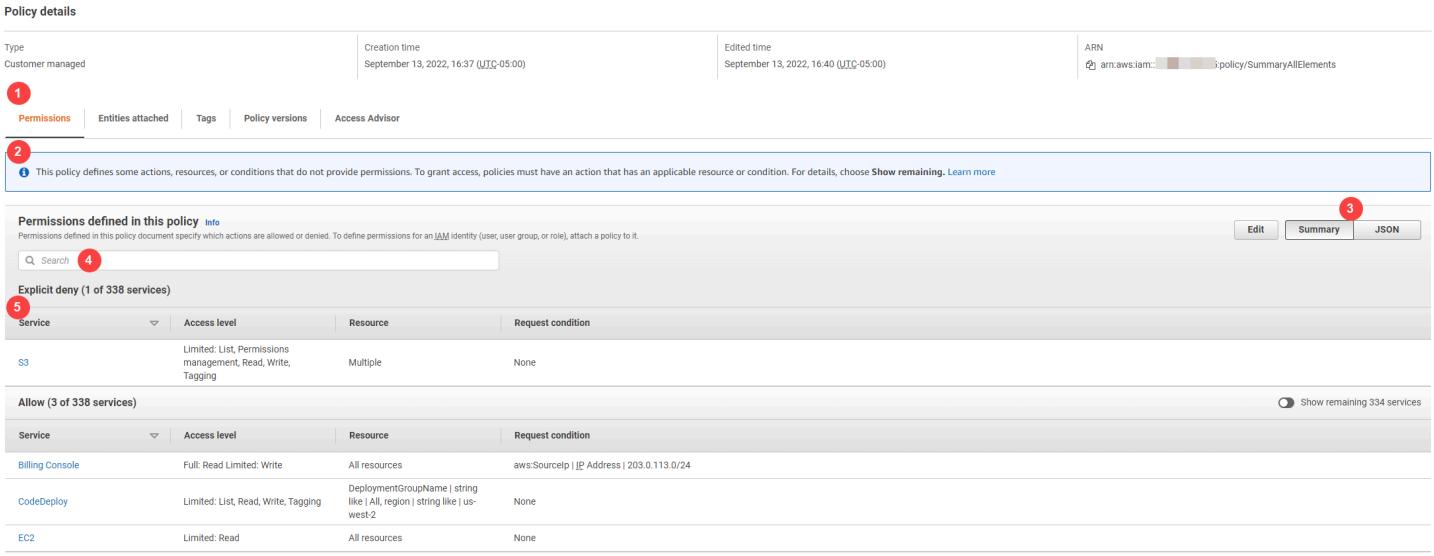
Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan kebijakan mencakup daftar layanan dan ringkasan izin yang ditentukan oleh kebijakan yang dipilih.



Tabel ringkasan kebijakan dikelompokkan menjadi satu atau lebih bagian Layanan yang tidak terkategorikan, Penolakan secara eksplisit , dan Izinkan. Jika kebijakan mencakup layanan yang IAM tidak dikenali, maka layanan tersebut termasuk dalam bagian Layanan Tak Berkategori pada tabel. Jika IAM mengenali layanan, maka itu termasuk di bawah bagian penolakan eksplisit atau Izinkan dari tabel, tergantung pada efek kebijakan (`Deny`atau`Allow`).

Memahami elemen ringkasan kebijakan

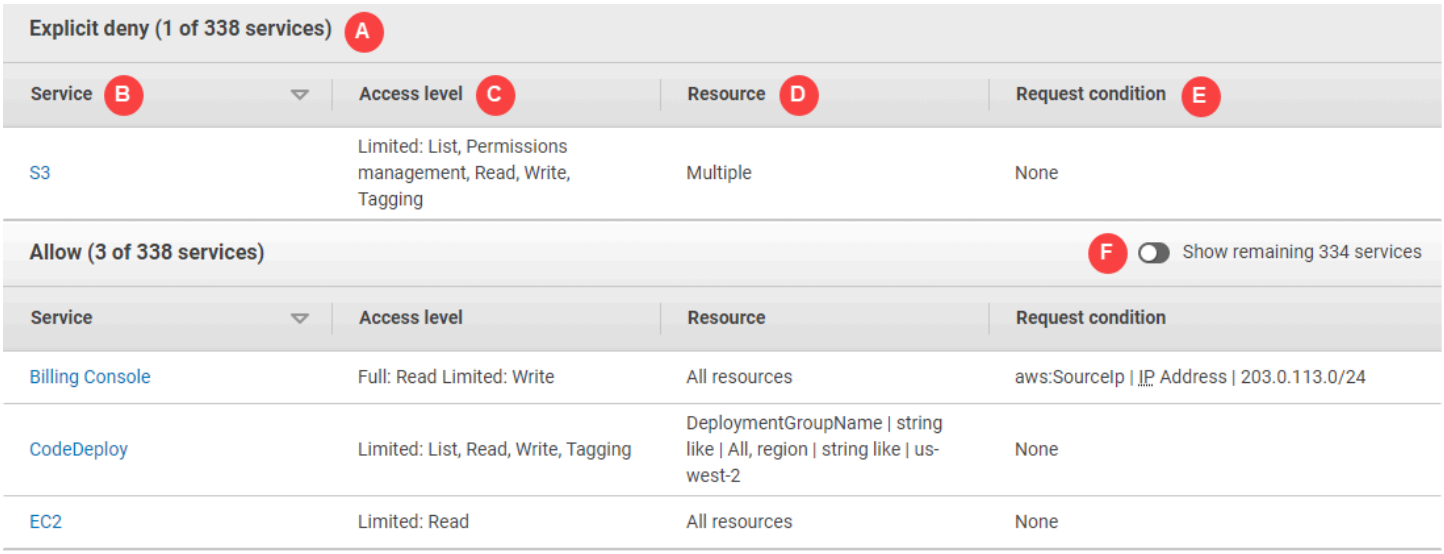
Dalam contoh halaman detail kebijakan berikut, kebijakan tersebut adalah SummaryAllElementskebijakan terkelola (kebijakan yang dikelola pelanggan) yang dilampirkan langsung ke pengguna. Kebijakan ini diperluas untuk menunjukkan ringkasan kebijakan.



Pada gambar sebelumnya, ringkasan kebijakan terlihat dari dalam halaman Kebijakan:

1. Tab Izin menyertakan izin yang ditentukan dalam kebijakan.
2. Jika kebijakan tidak memberikan izin untuk semua tindakan, sumber daya, dan syarat yang ditentukan dalam kebijakan, maka peringatan atau banner muncul di bagian atas halaman. Ringkasan kebijakan kemudian mencakup perincian tentang masalah. Untuk mempelajari bagaimana rangkuman kebijakan membantu Anda memahami dan mengatasi masalah izin yang diberikan oleh kebijakan Anda, lihat the section called “Kebijakan saya tidak memberikan izin yang diharapkan”.
3. Gunakan Ringkasan dan JSONtombol untuk beralih antara ringkasan kebijakan dan dokumen JSON kebijakan.
4. Gunakan kotak Pencarian untuk mengurangi daftar layanan dan menemukan layanan tertentu.
5. Tampilan yang diperluas menunjukkan rincian tambahan SummaryAllElementskebijakan.

Gambar tabel ringkasan kebijakan berikut menunjukkan SummaryAllElementskebijakan yang diperluas pada halaman detail kebijakan.



Explicit deny (1 of 338 services) A

Service B	Access level C	Resource D	Request condition E
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	None

Allow (3 of 338 services) F Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeDeploy	Limited: List, Read, Write, Tagging	DeploymentGroupName | string like | All, region | string like | us-west-2	None
EC2	Limited: Read	All resources	None

Pada gambar sebelumnya, ringkasan kebijakan terlihat dari dalam halaman Kebijakan:

A. Untuk layanan yang IAM mengakui, ia mengatur layanan sesuai dengan apakah kebijakan mengizinkan atau secara eksplisit menolak penggunaan layanan. Dalam contoh ini, kebijakan tersebut mencakup `Deny` pernyataan untuk layanan Amazon S3 dan `Allow` pernyataan untuk Layanan Penagihan,, CodeDeploy dan Amazon. EC2

B. Layanan – Kolom ini mencantumkan layanan yang ditegaskan dalam kebijakan dan menyajikan detail untuk setiap layanan. Setiap nama layanan dalam tabel ringkasan kebijakan adalah tautan ke ringkasan layanan yang dijelaskan dalam Ringkasan layanan (daftar tindakan). Dalam contoh ini, izin ditentukan untuk layanan Amazon S3, Penagihan CodeDeploy, dan Amazon. EC2

C. Tingkat akses — Kolom ini memberi tahu apakah tindakan di setiap tingkat akses (`List`,`Read`,`Write`,`Permission Management`,, dan`Tagging`) memiliki `Full` atau `Limited` izin yang ditentukan dalam kebijakan. Untuk detail dan contoh tambahan dari ringkasan tingkat akses, lihat Tingkat akses dalam ringkasan kebijakan.

- Akses penuh – Entri ini menunjukkan bahwa layanan memiliki akses ke semua tindakan, di dalamnya tersedia empat tingkat akses untuk layanan tersebut.
- Jika entri tidak mencakup Akses penuh, maka layanan memiliki akses ke beberapa tetapi tidak semua tindakan untuk layanan tersebut. Akses kemudian didefinisikan dengan deskripsi berikut untuk masing-masing klasifikasi tingkat akses (`List`,,, `Read` `WritePermission Management`, dan`Tagging`):

  Lengkap: Kebijakan ini menyediakan akses ke semua tindakan yang di dalamnya mencantumkan setiap klasifikasi tingkat akses. Dalam contoh ini, kebijakan ini memberikan akses ke semua `Read` aksi nyata.

Terbatas: Kebijakan ini menyediakan akses ke satu atau beberapa tetapi tidak semua tindakan tercantum dalam klasifikasi tingkat akses. Dalam contoh ini, kebijakan ini memberikan akses ke beberapa `Write` aksi nyata.

D. Sumber Daya – Kolom ini menunjukkan sumber daya yang ditentukan kebijakan untuk setiap layanan.

- Banyak – Kebijakan ini mencakup lebih dari satu, tetapi tidak semua sumber daya di balik layanan tersebut. Dalam contoh ini, akses secara eksplisit ditolak untuk lebih dari satu sumber daya Amazon S3
- Semua sumber daya — Kebijakan ini ditetapkan untuk semua sumber daya dalam layanan. Dalam contoh ini, kebijakan ini memungkinkan tindakan terdaftar untuk dilakukan pada semua sumber daya tagihan.
- Teks Sumber Daya – Kebijakan ini mencakup suatu sumber daya yang terdapat dalam layanan. Dalam contoh ini, tindakan yang tercantum hanya diperbolehkan pada `DeploymentGroupName` CodeDeploy sumber daya. Bergantung pada informasi yang disediakan layananIAM, Anda mungkin melihat ARN atau Anda mungkin melihat jenis sumber daya yang ditentukan.

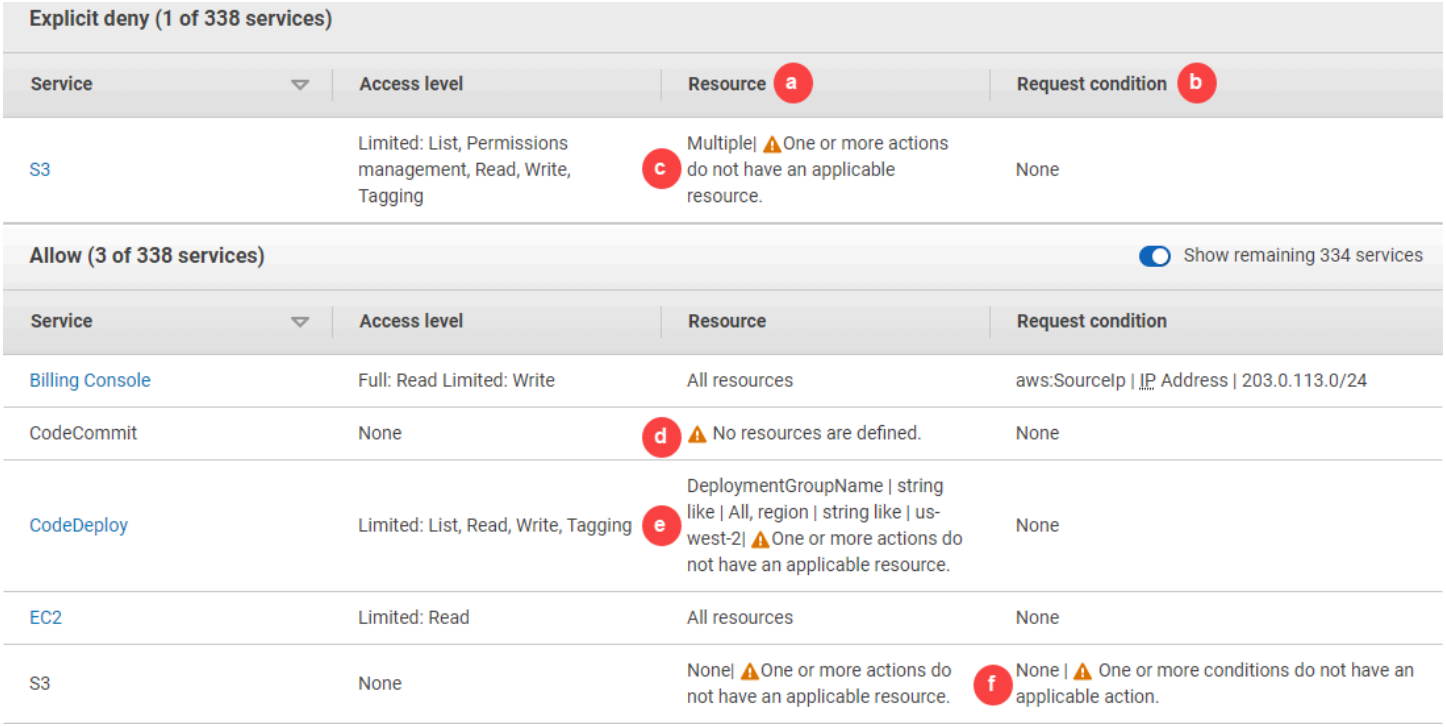
> **Note**
>
> Kolom ini dapat menyertakan sumber daya dari layanan yang berbeda. Jika pernyataan kebijakan yang mencakup sumber daya tidak mencakup tindakan dan sumber daya dari layanan yang sama, maka kebijakan Anda mencakup sumber daya yang tidak sesuai. IAMtidak memperingatkan Anda tentang sumber daya yang tidak cocok saat Anda membuat kebijakan, atau saat Anda melihat kebijakan dalam ringkasan kebijakan. Jika kolom ini memuat sumber daya yang tidak sesuai, maka Anda harus meninjau kebijakan Anda untuk kesalahan. Untuk lebih memahami kebijakan Anda, selalu uji kebijakan tersebut dengan simulator kebijakan.

E. Minta kondisional – Kolom ini menunjukkan apakah layanan atau tindakan yang terkait dengan sumber daya ditujukan kepada kondisi-kondisi.

- Tidak ada – Kebijakan ini tidak mengikutkan kondisi untuk layanan. Dalam contoh ini tidak ada kondisi yang diterapkan pada tindakan yang ditolak dalam layanan Amazon S3
- Kondisi teks – Kebijakan ini mencakup suatu kondisi untuk layanan. Dalam contoh ini, tindakan Penagihan yang terdaftar hanya diperbolehkan jika alamat IP sumber cocok`203.0.113.0/24`.
- Banyak – Kebijakan ini mencakup lebih dari satu kondisi untuk layanan tersebut. Untuk melihat masing-masing dari beberapa kondisi kebijakan, pilih JSONuntuk melihat dokumen kebijakan.

F. Tampilkan layanan yang tersisa — Alihkan tombol ini untuk memperluas tabel untuk menyertakan layanan yang tidak ditentukan oleh kebijakan. Layanan ini ditolak secara implisit (atau ditolak secara default) dalam kebijakan ini. Namun, pernyataan dalam kebijakan lain mungkin masih mengizinkan atau secara eksplisit menolak menggunakan layanan tersebut. Ringkasan kebijakan merangkum izin satu kebijakan. Untuk mempelajari tentang bagaimana AWS Layanan memutuskan apakah permintaan yang diberikan harus diizinkan atau ditolak, lihatLogika evaluasi kebijakan.

Jika kebijakan atau elemen dalam kebijakan tidak memberikan izin, IAM berikan peringatan dan informasi tambahan dalam ringkasan kebijakan. Tabel ringkasan kebijakan berikut menunjukkan perluasan Tampilkan layanan layanan yang tersisa di halaman detail SummaryAllElementskebijakan dengan kemungkinan peringatan.



Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

Dalam image sebelumnya, Anda dapat melihat semua layanan yang mencakup tindakan, sumber daya, atau kondisi tertentu, tanpa izin:

a. Peringatan sumber daya – Untuk layanan yang tidak memberikan izin untuk semua tindakan atau sumber daya yang disertakan, Anda melihat salah satu upaya berikut di kolom Sumber Daya pada tabel:

- Tidak ada sumber daya yang ditentukan. – Ini berarti bahwa layanan telah menentukan tindakan tetapi tidak ada sumber daya didukung yang tercakup dalam kebijakan.
- Satu atau lebih tindakan tidak memiliki sumber daya yang berlaku. – Ini berarti bahwa layanan telah menentukan tindakan, tetapi bahwa beberapa tindakan tersebut tidak memiliki sumber daya yang Support.
- Satu atau lebih sumber daya tidak memiliki tindakan yang berlaku. – Ini berarti bahwa layanan telah mendefinisikan sumber daya, tetapi beberapa sumber daya tersebut tidak memiliki tindakan yang mendukung.

Jika layanan mencakup kedua tindakan yang tidak memiliki sumber daya dan sumber daya yang berlaku yang memiliki sumber daya yang berlaku, maka hanya Satu atau lebih sumber daya yang tidak memiliki tindakan yang berlaku. peringatan ditampilkan. Ini karena ketika Anda melihat ringkasan layanan untuk layanan tersebut, sumber daya yang tidak berlaku untuk tindakan apa pun tidak ditampilkan. Untuk tindakan `ListAllMyBuckets`, kebijakan ini mencakup peringatan terakhir karena tindakan tersebut tidak mendukung izin tingkat sumber daya, dan tidak mendukung tombol syarat `s3:x-amz-acl`. Jika Anda memperbaiki masalah sumber daya atau masalah syarat, masalah yang tersisa muncul di peringatan mendetail.

b. Minta peringatan kondisional – Untuk layanan yang tidak memberikan izin untuk semua kondisi yang disertakan, Anda melihat salah satu peringatan berikut dalam kolom Minta kondisional pada tabel:

- Satu atau lebih tindakan tidak memiliki kondisi yang berlaku. – Ini berarti bahwa layanan telah menentukan tindakan, tetapi beberapa tindakan tersebut tidak memiliki kondisi yang mendukung
- Satu atau lebih kondisi tidak memiliki tindakan yang berlaku. – Ini berarti bahwa layanan telah memiliki kondisi yang ditetapkan, tetapi sebagian dari kondisi tersebut tidak memiliki tindakan yang mendukung.

c. Beberapa |

Satu atau lebih tindakan tidak memiliki sumber daya yang berlaku. – `Deny` pernyataan untuk

Amazon S3 mencakup lebih dari satu sumber daya. Ia juga mencakup lebih dari satu tindakan, dan beberapa tindakan mendukung sumber daya dan beberapa lainnya tidak. Untuk melihat kebijakan ini, lihat the section called “SummaryAllElementsJSONdokumen kebijakan”. Dalam hal ini, kebijakan ini mencakup semua tindakan Amazon S3 dan hanya tindakan yang dapat dilakukan pada bucket atau objek bucket yang ditolak.

d. ⚠ Tidak ada sumber daya yang ditentukan — Layanan memiliki tindakan yang ditentukan, tetapi tidak ada sumber daya yang didukung yang disertakan dalam kebijakan, dan oleh karena itu layanan tidak memberikan izin. Dalam hal ini, kebijakan mencakup CodeCommit tindakan tetapi tidak ada CodeCommit sumber daya.

e. DeploymentGroupName | string seperti | Semua, wilayah | string seperti | us-west-2 ⚠ | Satu atau lebih tindakan tidak memiliki sumber daya yang berlaku. — Layanan memiliki tindakan yang ditentukan, dan setidaknya satu tindakan lagi yang tidak memiliki sumber daya pendukung.

f. Tidak ada | ⚠ Satu atau lebih kondisi tidak memiliki tindakan yang berlaku. — Layanan ini memiliki setidaknya satu kunci kondisi yang tidak memiliki tindakan pendukung.

SummaryAllElementsJSONdokumen kebijakan

SummaryAllElementsKebijakan ini tidak dimaksudkan untuk Anda gunakan untuk menentukan izin di akun Anda. Sebaliknya, ia disertakan untuk menunjukkan kesalahan dan peringatan yang mungkin Anda temukan saat melihat ringkasan kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
```

```
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-project"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
```

```
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
                    ]
                }
            }
        }
    ]
}
```

Lihat ringkasan kebijakan

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke IAM pengguna atau peran. Untuk kebijakan terkelola, Anda dapat melihat ringkasan kebijakan di halaman Kebijakan. Jika kebijakan Anda tidak menyertakan ringkasan kebijakan, lihat Ringkasan kebijakan hilang untuk mempelajari alasannya.

Melihat ringkasan kebijakan dari halaman Kebijakan

Anda dapat melihat ringkasan kebijakan untuk kebijakan yang dikelola pada halaman Kebijakan.

Untuk melihat ringkasan kebijakan dari halaman Kebijakan

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.

4. Pada halaman Detail kebijakan untuk kebijakan, lihat tab Izin untuk melihat ringkasan kebijakan.

Melihat ringkasan kebijakan untuk kebijakan yang dilampirkan ke pengguna

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke IAM pengguna.

Untuk melihat ringkasan kebijakan yang terlampir pada pengguna

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat.
4. Di Ringkasan untuk pengguna, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada pengguna secara langsung atau dari grup.
5. Dalam tabel kebijakan untuk pengguna, perluas baris kebijakan yang ingin Anda lihat.

Melihat ringkasan kebijakan untuk kebijakan yang dilampirkan pada peran

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke peran.

Untuk melihat ringkasan kebijakan yang terlampir pada peran

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Dalam daftar peran, pilih nama peran yang kebijakannya ingin Anda lihat.
4. Di Ringkasan untuk peran, lihat Izin untuk melihat daftar kebijakan yang terlampir pada peran tersebut.
5. Dalam tabel kebijakan untuk peran tersebut, perluas baris kebijakan yang ingin Anda lihat.

Menyunting kebijakan untuk memperbaiki peringatan

Saat melihat ringkasan kebijakan, Anda mungkin menemukan kesalahan ketik atau pemberitahuan bahwa kebijakan tersebut tidak memberikan izin yang Anda harapkan. Anda tidak dapat langsung menyunting ringkasan kebijakan. Namun demikian, Anda dapat menyunting kebijakan terkelola pelanggan dengan menggunakan editor kebijakan visual, yang menangkap banyak kesalahan dan

peringatan yang dilaporkan oleh ringkasan kebijakan. Kemudian, Anda dapat melihat perubahan dalam ringkasan kebijakan untuk mengonfirmasi bahwa Anda sudah menyelesaikan semua masalah. Untuk mempelajari cara sunting kebijakan selaras, lihat the section called “Edit IAM kebijakan”. Anda tidak dapat mengedit AWS kebijakan terkelola.

Anda dapat mengedit kebijakan untuk ringkasan kebijakan Anda menggunakan opsi Visual.

Untuk mengedit kebijakan ringkasan kebijakan Anda menggunakan opsi Visual

1. Buka ringkasan kebijakan seperti yang dijelaskan dalam prosedur sebelumnya.
2. Pilih Edit.

   Jika Anda berada di halaman Pengguna dan memilih untuk menyunting kebijakan terkelola pelanggan yang terlampir pada pengguna tersebut, Anda akan diarahkan kembali ke halaman Kebijakan. Anda dapat menyunting kebijakan yang terkelola pelanggan hanya di halaman Kebijakan.
3. Pilih opsi Visual untuk melihat representasi visual kebijakan yang dapat diedit. IAMmungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual dan untuk membuatnya lebih mudah bagi Anda untuk menemukan dan memperbaiki masalah. Peringatan dan pesan adanya kesalahan di halaman dapat memandu Anda untuk membenahi adanya masalah dalam kebijakan Anda. Untuk informasi selengkapnya tentang cara IAM merestrukturisasi kebijakan, lihatRestrukturisasi kebijakan.
4. Edit kebijakan Anda dan pilih Berikutnya untuk melihat perubahan yang tercermin dalam ringkasan kebijakan. Jika Anda masih melihat masalah, pilih Sebelumnya untuk kembali ke layar penyuntingan.
5. Pilih Simpan perubahan untuk menyimpan perubahan Anda.

Anda dapat mengedit kebijakan untuk ringkasan kebijakan Anda menggunakan JSONopsi.

Untuk mengedit kebijakan ringkasan kebijakan Anda menggunakan JSONopsi

1. Buka ringkasan kebijakan seperti yang dijelaskan dalam prosedur sebelumnya.
2. Anda dapat menggunakan Ringkasan dan JSONtombol untuk membandingkan ringkasan kebijakan dengan dokumen JSON kebijakan. Anda dapat menggunakan informasi ini untuk menentukan baris mana dalam dokumen kebijakan yang ingin Anda ubah.
3. Pilih Edit dan kemudian pilih JSONopsi untuk mengedit dokumen JSON kebijakan.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di opsi Editor visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

Jika Anda berada di halaman Pengguna dan memilih untuk menyunting kebijakan terkelola pelanggan yang terlampir pada pengguna tersebut, Anda akan diarahkan kembali ke halaman Kebijakan. Anda dapat menyunting kebijakan yang terkelola pelanggan hanya di halaman Kebijakan.

4. Edit kebijakan Anda. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya. Jika Anda masih melihat masalah, pilih Sebelumnya untuk kembali ke layar penyuntingan.
5. Pilih Simpan perubahan untuk menyimpan perubahan Anda.

Tingkat akses dalam ringkasan kebijakan

AWS ringkasan tingkat akses

Ringkasan kebijakan mencakup ringkasan tingkat akses yang menjelaskan izin tindakan yang ditentukan untuk setiap layanan yang disebutkan dalam kebijakan. Untuk ARN ringkasan kebijakan, lihat Ringkasan kebijakan. Ringkasan tingkat akses menunjukkan apakah tindakan di setiap tingkat akses (`List`,,, `Read` `TaggingWrite`, dan`Permissions management`) memiliki `Full` atau `Limited` izin yang ditentukan dalam kebijakan. Untuk melihat klasifikasi tingkat akses yang ditetapkan untuk setiap tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Contoh berikut menjelaskan akses yang diberikan oleh kebijakan untuk layanan tertentu. Untuk contoh dokumen JSON kebijakan lengkap dan ringkasan terkait, lihat. Contoh ringkasan kebijakan

Layanan	Tingkat akses	Kebijakan memungkinkan hal-hal berikut
IAM	Akses penuh	Akses ke semua tindakan dalam IAM layanan.

Layanan	Tingkat akses	Kebijakan memungkinkan hal-hal berikut
CloudWatch	Penuh:Daftar	Akses ke semua CloudWatch tindakan di tingkat `List` akses, tetapi tidak ada akses ke tindakan dengan`Read`,`Write`, atau klasifikasi tingkat `Permissions management` akses.
Data Pipeline	Terbatas: Daftar, Baca	Akses ke setidaknya satu tetapi tidak semua AWS Data Pipeline tindakan di `List` dan tingkat `Read` akses, tetapi tidak `Write` atau `Permissions management` tindakan.
EC2	Penuh:Daftar, Baca Terbatas: Tulis	Akses ke semua Amazon EC2 `List` dan `Read` tindakan dan akses ke setidaknya satu tetapi tidak semua EC2 `Write` tindakan Amazon, tetapi tidak ada akses ke tindakan dengan klasifikasi tingkat `Permissions management` akses.
S3	Terbatas: Manajemen Baca, Tulis, Izin	Akses ke setidaknya satu tetapi tidak semua Amazon S3 `Read`, `Write` dan tindakan `Permissions management` .
CodeDeploy	(kosong)	Akses tidak dikenal, karena IAM tidak mengenali layanan ini.
APIGerbang	Tidak ada	Tidak ada akses yang didefinisikan dalam kebijakan.
CodeBuild	⚠ Tidak ada tindakan yang ditentukan.	Tidak ada akses karena tidak ada tindakan yang ditentukan untuk layanan tersebut. Untuk mempelajari cara memahami dan mengatasi masalah ini, lihat the section called “Kebijakan saya tidak memberikan izin yang diharapkan”.

Dalam ringkasan kebijakan, Akses penuh menunjukkan bahwa kebijakan menyediakan akses ke semua tindakan dalam layanan. Kebijakan yang memberikan akses ke beberapa tetapi tidak semua

tindakan dalam layanan akan dikelompokkan lebih lanjut sesuai dengan klasifikasi tingkat akses. Hal ini ditunjukkan oleh salah satu pengelompokan tingkat akses berikut:

- Penuh: Kebijakan tersebut memberikan akses ke semua tindakan dalam klasifikasi tingkat akses yang ditentukan.
- Terbatas: Kebijakan tersebut memberikan akses ke satu atau lebih tetapi tidak semua tindakan dalam klasifikasi tingkat akses yang ditentukan.
- TIdak Ada: Kebijakan ini tidak memiliki akses.
- (kosong): IAM tidak mengenali layanan ini. Jika nama layanan mencakup salah ketik, maka kebijakan tersebut tidak memberikan akses ke layanan. Jika nama layanan sudah benar, maka layanan mungkin tidak mendukung rangkuman kebijakan atau mungkin sedang ditampilkan di pratinjau. Dalam hal ini, kebijakan mungkin memberikan akses, tetapi akses tersebut tidak dapat ditampilkan dalam ringkasan kebijakan. Untuk meminta dukungan ringkasan kebijakan untuk layanan yang tersedia secara umum (GA), lihat Layanan tidak mendukung ringkasan IAM kebijakan.

Ringkasan tingkat akses yang mencakup akses terbatas (sebagian) ke tindakan dikelompokkan menggunakan klasifikasi tingkat AWS aksesList,,,Read, Tagging atau. Write Permissions management

AWS tingkat akses

AWS mendefinisikan klasifikasi tingkat akses berikut untuk tindakan dalam layanan:

- Daftar: Izin untuk mencantumkan sumber daya di dalam layanan untuk menentukan apakah ada objek. Tindakan dengan tingkat akses ini dapat mencantumkan objek tetapi tidak dapat melihat isi sumber daya. Misalnya, tindakan Amazon S3 `ListBucket` memiliki tingkat akses Daftar.
- Baca: Izin untuk membaca, namun tidak mengedit konten dan atribut sumber daya dalam layanan. Misalnya, tindakan Amazon S3 `GetObject` and `GetBucketLocation` memiliki tingkat akses Baca.
- Penandaan: Izin untuk melakukan tindakan yang hanya mengubah status tanda sumber daya. Misalnya, IAM tindakan `TagRole` dan `UntagRole` memiliki tingkat akses Tagging karena hanya mengizinkan penandaan atau pembatalan tag peran. Namun, tindakan `CreateRole` memungkinkan menandai sumber daya peran saat Anda membuat peran tersebut. Karena tindakan tersebut tidak hanya menambahkan tanda, tindakan memiliki tingkat akses `Write`.

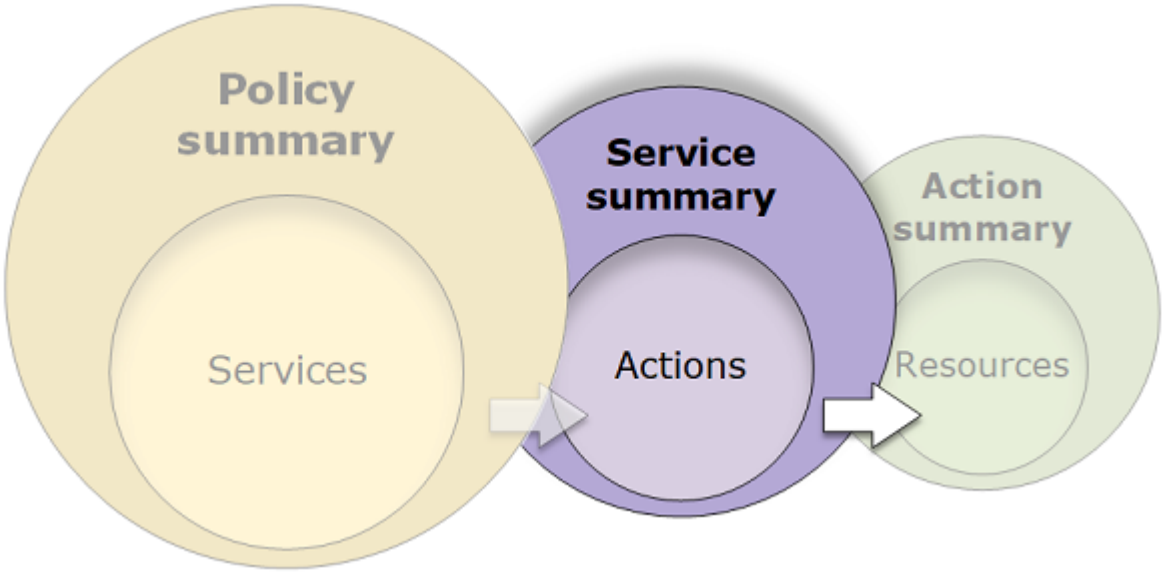
- Tulis: Izin untuk membuat, menghapus, atau memodifikasi sumber daya dalam layanan. Misalnya, tindakan Amazon S3`CreateBucket`, `DeleteBucket` dan `PutObject` memiliki tingkat akses Tulis. `Write`tindakan mungkin juga memungkinkan memodifikasi tag sumber daya. Namun, tindakan yang hanya memungkinkan perubahan pada tanda memiliki tingkat akses `Tagging`.
- Manajemen izin Izin untuk memberikan atau mengubah izin sumber daya dalam layanan. Misalnya, sebagian besar IAM dan AWS Organizations tindakan, serta tindakan seperti tindakan Amazon S3 `PutBucketPolicy` dan `DeleteBucketPolicy` memiliki tingkat akses manajemen Izin.

> **Kiat**
>
> Untuk meningkatkan keamanan Anda Akun AWS, batasi atau pantau secara teratur kebijakan yang mencakup klasifikasi tingkat akses manajemen izin.

Untuk melihat klasifikasi tingkat akses untuk semua tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Ringkasan layanan (daftar tindakan)

Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan layanan menyertakan daftar tindakan dan ringkasan izin yang ditentukan oleh kebijakan untuk layanan yang dipilih.

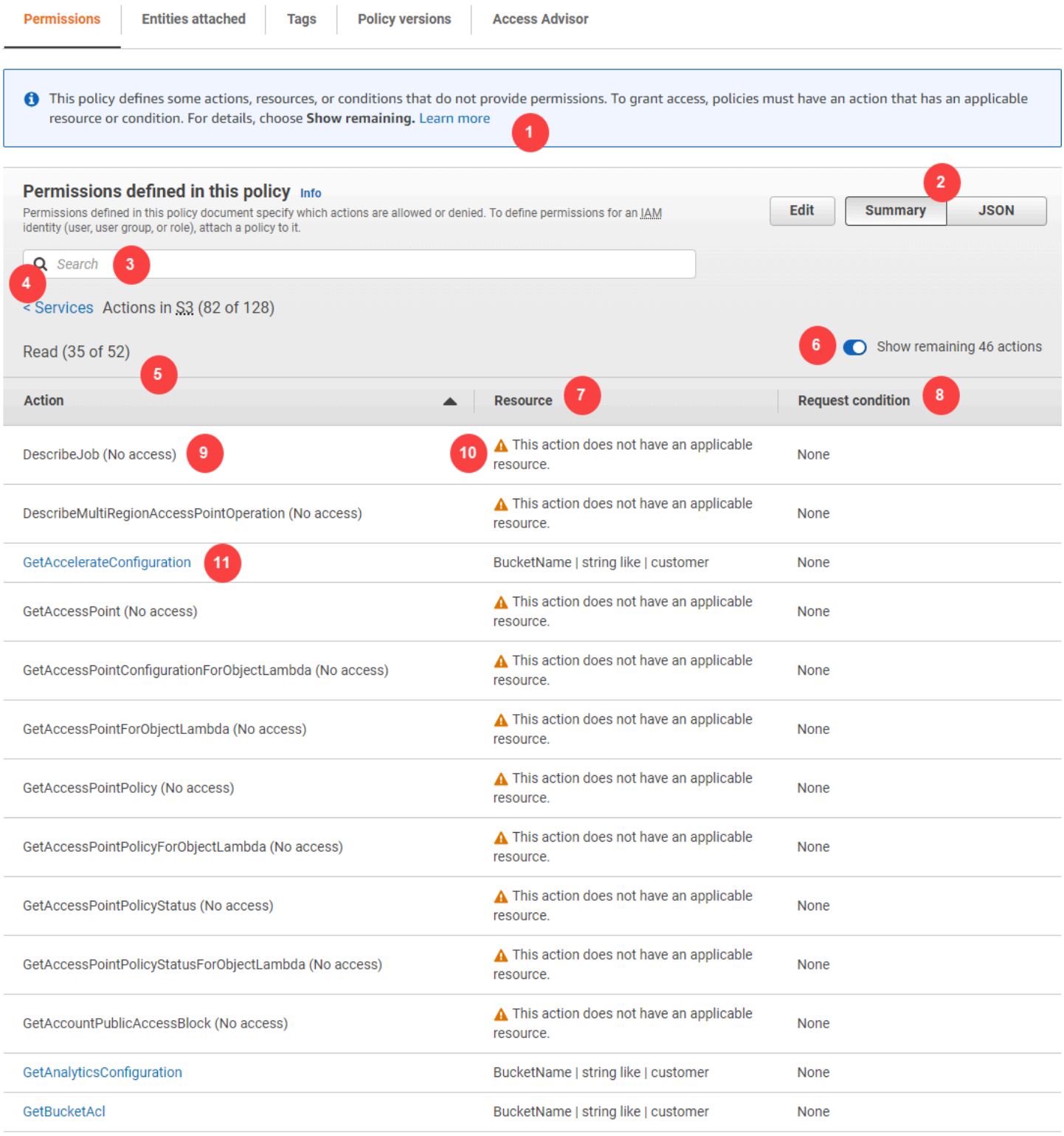


Anda dapat melihat ringkasan layanan untuk setiap layanan yang tercantum dalam ringkasan kebijakan yang memberikan izin. Tabel dikelompokkan ke dalam Tindakan yang tidak dikategorikan,

Jenis sumber daya yang tidak dikategorikan, dan bagian tingkat akses. Jika kebijakan mencakup tindakan yang IAM tidak dikenali, maka tindakan tersebut disertakan dalam bagian Tindakan Tidak Dikategorikan pada tabel. Jika IAM mengenali tindakan, maka itu termasuk di bawah salah satu bagian tingkat akses (Daftar, Baca, Tulis, dan Manajemen Izin) dari tabel. Untuk melihat klasifikasi tingkat akses yang ditetapkan untuk setiap tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Memahami elemen ringkasan layanan

Contoh di bawah ini adalah ringkasan layanan untuk tindakan Amazon S3 yang diizinkan dari ringkasan kebijakan. Tindakan untuk layanan ini dikelompokkan berdasarkan tingkat akses. Misalnya, 35 tindakan Baca didefinisikan dari total 52 tindakan Baca yang tersedia untuk layanan.



Halaman ringkasan layanan untuk kebijakan yang dikelola mencakup informasi berikut ini:

1. Jika kebijakan tidak memberikan izin untuk semua tindakan, sumber daya, dan kondisi yang ditentukan untuk layanan dalam kebijakan, banner peringatan akan muncul di bagian atas

halaman. Ringkasan layanan kemudian mencakup rincian tentang masalah. Untuk mempelajari bagaimana rangkuman kebijakan membantu Anda memahami dan mengatasi masalah izin yang diberikan oleh kebijakan Anda, lihat the section called “Kebijakan saya tidak memberikan izin yang diharapkan”.

2. Pilih JSONuntuk melihat detail tambahan tentang kebijakan tersebut. Anda dapat melakukan ini untuk melihat semua kondisi yang diterapkan ke tindakan. (Jika Anda melihat ringkasan layanan untuk kebijakan sebaris yang dilampirkan langsung ke pengguna, Anda harus menutup kotak dialog ringkasan layanan dan kembali ke ringkasan kebijakan untuk mengakses dokumen JSON kebijakan.)
3. Untuk melihat ringkasan tindakan tertentu, ketik kata kunci ke dalam kotak Pencarian untuk mengurangi daftar tindakan yang tersedia.
4. Di sebelah panah belakang Layanan muncul nama layanan (dalam hal ini S3). Ringkasan layanan untuk layanan ini mencakup daftar tindakan yang diizinkan atau ditolak yang didefinisikan dalam kebijakan. Jika layanan muncul di bawah (Penolakan eksplisit) pada tab Izin, maka tindakan yang tercantum dalam tabel ringkasan layanan ditolak secara eksplisit. Jika layanan muncul di bawah Izinkan pada tab Izin, maka tindakan yang tercantum dalam tabel ringkasan layanan diizinkan.
5. Tindakan - Kolom ini mencantumkan tindakan yang ditentukan dalam kebijakan dan menyediakan sumber daya dan kondisi untuk setiap tindakan. Jika kebijakan memberikan atau menolak izin untuk tindakan, maka nama tindakan akan ditautkan ke tabel ringkasan tindakan. Tabel mengelompokkan tindakan ini menjadi setidaknya satu atau hingga lima bagian, tergantung pada tingkat akses yang diizinkan atau ditolak oleh kebijakan tersebut. Bagian-bagiannya adalah Daftar, Baca, Tulis, Manajemen Izin, dan Penandaan. Hitungan menunjukkan jumlah tindakan yang diakui yang memberikan izin dalam setiap tingkat akses. Totalnya adalah jumlah tindakan yang diketahui untuk layanan tersebut. Dalam contoh ini, 35 tindakan memberikan izin dari 52 total tindakan Amazon S3 Read yang diketahui. Untuk melihat klasifikasi tingkat akses yang ditetapkan untuk setiap tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.
6. Tampilkan tindakan yang tersisa — Alihkan tombol ini untuk memperluas atau menyembunyikan tabel untuk menyertakan tindakan yang diketahui tetapi tidak memberikan izin untuk layanan ini. Mengaktifkan tombol juga menampilkan peringatan untuk elemen apa pun yang tidak memberikan izin.
7. Sumber Daya – Kolom ini menunjukkan sumber daya yang ditentukan oleh kebijakan untuk layanan. IAMtidak memeriksa apakah sumber daya berlaku untuk setiap tindakan. Dalam contoh ini, tindakan dalam layanan Amazon S3 hanya diperbolehkan pada sumber daya bucket Amazon `developer_bucket` S3. Bergantung pada informasi yang disediakan layananIAM, Anda mungkin

melihat ARN seperti`arn:aws:s3:::developer_bucket/*`, atau Anda mungkin melihat jenis sumber daya yang ditentukan, seperti`BucketName = developer_bucket`.

> **Note**
>
> Kolom ini dapat menyertakan sumber daya dari layanan yang berbeda. Jika pernyataan kebijakan yang mencakup sumber daya tidak mencakup tindakan dan sumber daya dari layanan yang sama, maka kebijakan Anda mencakup sumber daya yang tidak sesuai. IAMtidak memperingatkan Anda tentang sumber daya yang tidak cocok saat Anda membuat kebijakan, atau saat Anda melihat kebijakan dalam ringkasan layanan. IAMjuga tidak menunjukkan apakah tindakan tersebut berlaku untuk sumber daya, hanya apakah layanan cocok. Jika kolom ini memuat sumber daya yang tidak sesuai, maka Anda harus meninjau kebijakan Anda untuk kesalahan. Untuk lebih memahami kebijakan Anda, selalu uji kebijakan tersebut dengan simulator kebijakan.

8. Minta ketentuan – Kolom ini memberi tahu apakah tindakan yang terkait dengan sumber daya tunduk pada kondisi. Untuk mempelajari lebih lanjut tentang kondisi tersebut, pilih JSONuntuk meninjau dokumen JSON kebijakan.
9. (Tanpa akses) – Kebijakan ini mencakup tindakan yang tidak menyediakan izin.
10. Peringatan sumber daya – Untuk tindakan dengan sumber daya yang tidak memberikan izin penuh, Anda akan melihat salah satu dari peringatan berikut:
    - Tindakan ini tidak mendukung izin tingkat sumber daya. Ini memerlukan wildcard (*) untuk sumber daya. – Artinya, kebijakan tersebut menyertakan izin tingkat sumber daya, tetapi harus menyertakan `"Resource": ["*"]` untuk memberikan izin untuk tindakan ini.
    - Tindakan ini tidak memiliki sumber daya yang berlaku. - Artinya tindakan tersebut termasuk dalam kebijakan tanpa sumber daya yang didukung.
    - Tindakan ini tidak memiliki sumber daya dan kondisi yang berlaku. - Artinya tindakan tersebut termasuk dalam kebijakan tanpa sumber daya yang didukung dan tanpa kondisi yang didukung. Dalam hal ini, ada juga ketentuan yang termasuk dalam kebijakan untuk layanan ini, tetapi tidak ada ketentuan yang berlaku untuk tindakan ini.
11. Tindakan yang memberikan izin meliputi tautan ke ringkasan tindakan.

Lihat ringkasan layanan

Anda dapat melihat ringkasan layanan untuk setiap layanan yang tercantum dalam ringkasan kebijakan yang memberikan izin.

Melihat ringkasan layanan dari halaman Kebijakan

Anda dapat melihat ringkasan layanan untuk kebijakan terkelola di halaman Kebijakan.

Untuk melihat ringkasan layanan untuk kebijakan yang dikelola

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
4. Pada halaman Detail kebijakan untuk kebijakan, lihat tab Izin untuk melihat ringkasan kebijakan.
5. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

Melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna

Anda dapat melihat ringkasan layanan untuk setiap kebijakan yang dilampirkan ke IAM pengguna.

Untuk melihat ringkasan layanan untuk kebijakan yang terlampir pada pengguna

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat.
4. Di Ringkasan untuk pengguna, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada pengguna secara langsung atau dari grup.
5. Dalam tabel kebijakan untuk pengguna, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Pengguna dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Pilih Ringkasan. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

> **Note**
>
> Jika kebijakan yang Anda pilih adalah kebijakan inline yang dilampirkan langsung ke pengguna, maka tabel ringkasan layanan muncul. Jika kebijakan tersebut merupakan kebijakan sebaris yang dilampirkan dari grup, maka Anda akan dibawa ke dokumen

JSON kebijakan untuk grup tersebut. Jika kebijakan tersebut adalah kebijakan yang dikelola, maka Anda dibawa ke ringkasan layanan untuk kebijakan tersebut di halaman Kebijakan.

Melihat ringkasan layanan untuk kebijakan yang dilampirkan pada peran

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke peran.

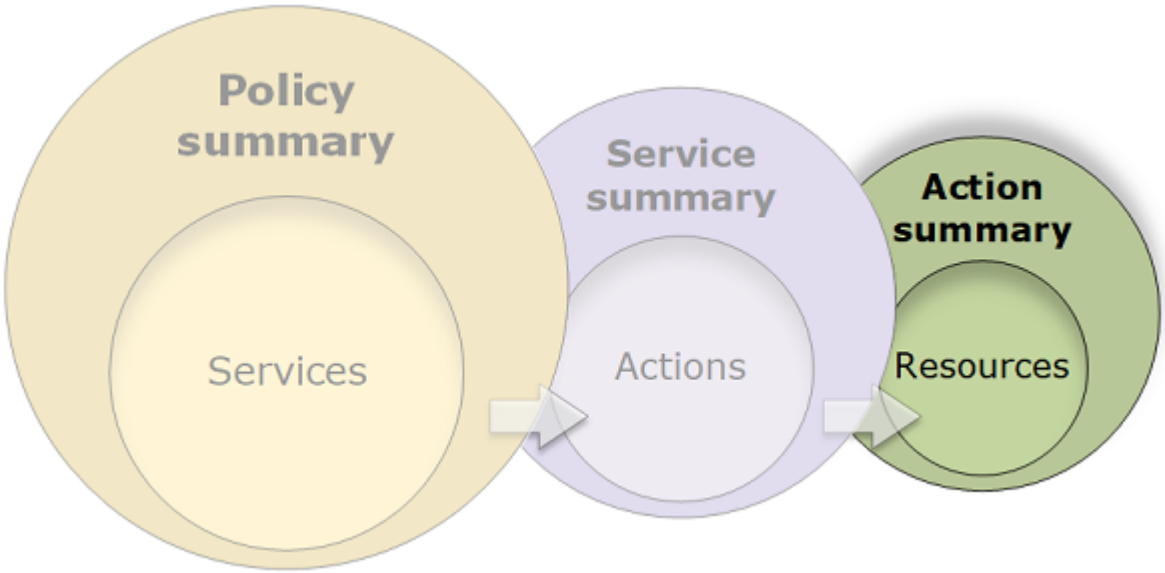
Untuk melihat ringkasan layanan untuk kebijakan yang terlampir pada peran

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Peran dari panel navigasi.
3. Dalam daftar peran, pilih nama peran yang kebijakannya ingin Anda lihat.
4. Di Ringkasan untuk peran, lihat Izin untuk melihat daftar kebijakan yang terlampir pada peran tersebut.
5. Dalam tabel kebijakan untuk peran tersebut, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Peran dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

Ringkasan tindakan (daftar sumber daya)

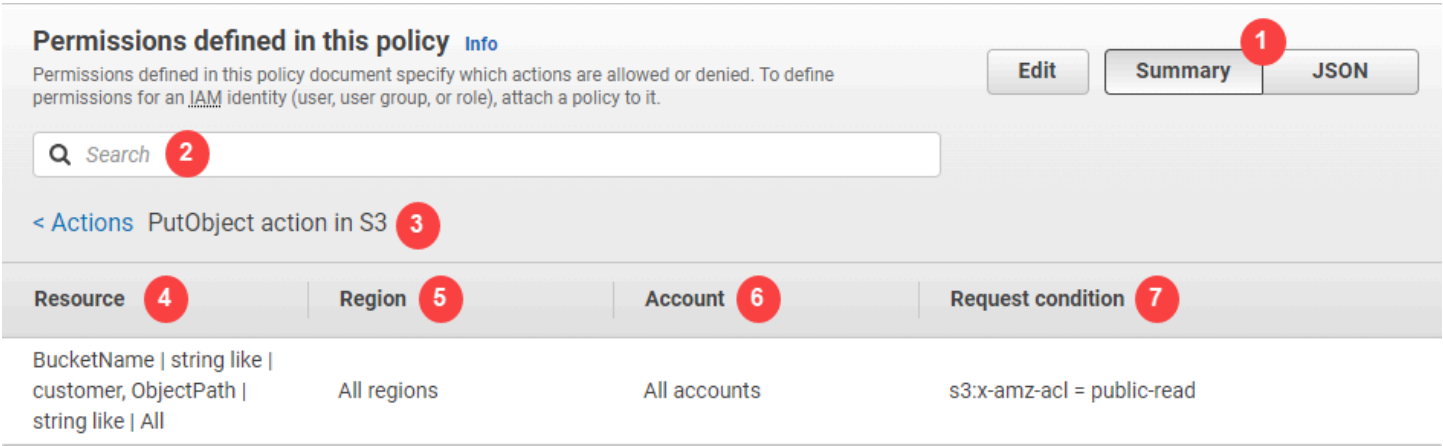
Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan tindakan mencakup daftar sumber daya dan ketentuan terkait yang berlaku untuk tindakan yang dipilih.



Untuk melihat ringkasan tindakan untuk setiap tindakan yang memberikan izin, pilih tautan dalam ringkasan layanan. Tabel ringkasan tindakan mencakup rincian tentang sumber daya, termasuk Wilayah dan Akunnya. Anda juga dapat melihat ketentuan yang berlaku untuk setiap sumber daya. Hal ini menunjukkan Anda ketentuan yang berlaku untuk beberapa sumber daya tetapi tidak yang lainnya.

Memahami elemen ringkasan tindakan

Contoh di bawah ini adalah ringkasan tindakan untuk tindakan `PutObject` (Tulis) dari ringkasan layanan Amazon S3 (lihat Ringkasan layanan (daftar tindakan)). Untuk tindakan ini, kebijakan ini mendefinisikan beberapa ketentuan pada satu sumber daya.



Halaman ringkasan tindakan mencakup informasi berikut:

1. Pilih JSONuntuk melihat detail tambahan tentang kebijakan, seperti melihat beberapa kondisi yang diterapkan pada tindakan. (Jika Anda melihat ringkasan tindakan untuk kebijakan sebaris yang dilampirkan langsung ke pengguna, langkah-langkahnya berbeda. Untuk mengakses dokumen JSON kebijakan dalam kasus itu, Anda harus menutup kotak dialog ringkasan tindakan dan kembali ke ringkasan kebijakan.)
2. Untuk melihat ringkasan sumber daya tertentu, ketik kata kunci ke dalam kotak Pencarian untuk mengurangi daftar sumber daya yang tersedia.
3. Di sebelah Tindakan panah kembali muncul nama layanan dan tindakan dalam format `action name action in service` (dalam hal ini PutObject tindakan di S3). Ringkasan tindakan untuk layanan ini mencakup daftar sumber daya yang ditetapkan dalam kebijakan.
4. Sumber Daya – Kolom ini mencantumkan sumber daya yang ditentukan kebijakan untuk layanan yang dipilih. Dalam contoh ini, PutObjecttindakan diizinkan di semua jalur objek, tetapi hanya pada sumber daya bucket `developer_bucket` Amazon S3. Bergantung pada informasi yang disediakan layananIAM, Anda mungkin melihat ARN sepertiarn:aws:s3:::developer_bucket/*, atau Anda mungkin melihat jenis sumber daya yang ditentukan, sepertiBucketName = developer_bucket, ObjectPath = All.
5. Wilayah – Kolom ini menunjukkan Wilayah tempat sumber daya ditetapkan. Sumber daya dapat ditentukan untuk semua Wilayah, atau satu Wilayah. Sumber daya tidak bisa ada di lebih dari satu Wilayah tertentu.
   - Semua wilayah — Tindakan yang terkait dengan sumber daya berlaku untuk semua Wilayah. Dalam contoh ini, tindakan tersebut menjadi milik layanan global, Amazon S3. Tindakan yang termasuk dalam layanan global berlaku untuk semua Wilayah.
   - Teks area – Tindakan terkait dengan sumber daya berlaku untuk satu Wilayah. Misalnya, kebijakan dapat menentukan WIlayah `us-east-2` untuk sumber daya.
6. Akun – Kolom ini menunjukkan apakah layanan atau tindakan yang terkait dengan sumber daya berlaku untuk akun tertentu. Sumber daya dapat muncul di semua akun atau satu akun. Sumber daya tidak bisa ada di lebih dari satu akun tertentu.
   - Semua akun – Tindakan yang terkait dengan sumber daya berlaku untuk semua akun. Dalam contoh ini, tindakan tersebut menjadi milik layanan global, Amazon S3. Tindakan yang termasuk dalam layanan global berlaku untuk semua akun.
   - Akun ini — Tindakan yang terkait dengan sumber daya hanya berlaku di akun saat ini..
   - Nomor akun – Tindakan yang terkait dengan sumber daya berlaku untuk satu akun (akun yang saat ini Anda tidak masuk). Misalnya, jika suatu kebijakan menetapkan akun `123456789012` untuk sumber daya, maka nomor akun muncul di ringkasan kebijakan.

7. Minta ketentuan – Kolom ini menunjukkan apakah tindakan yang terkait dengan sumber daya tunduk pada kondisi. Contoh ini mencakup ketentuan `s3:x-amz-acl = public-read`. Untuk mempelajari lebih lanjut tentang kondisi tersebut, pilih JSONuntuk meninjau dokumen JSON kebijakan.

Lihat ringkasan tindakan

Anda dapat melihat ringkasan tindakan untuk setiap tindakan yang tercantum dalam ringkasan kebijakan yang memberikan izin.

Melihat ringkasan tindakan dari halaman Kebijakan

Anda dapat melihat ringkasan tindakan untuk kebijakan terkelola.

Untuk melihat ringkasan tindakan untuk kebijakan yang dikelola

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
4. Pada halaman Detail kebijakan untuk kebijakan, lihat tab Izin untuk melihat ringkasan kebijakan.
5. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.
6. Di daftar tindakan ringkasan layanan, pilih nama tindakan yang ingin Anda lihat.

Melihat ringkasan tindakan untuk kebijakan yang dilampirkan ke pengguna

Anda dapat melihat ringkasan tindakan untuk kebijakan apa pun yang dilampirkan ke pengguna.

Untuk melihat ringkasan tindakan untuk kebijakan yang terlampir pada pengguna

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat.
4. Di Ringkasan untuk pengguna, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada pengguna secara langsung atau dari grup.

5. Dalam tabel kebijakan untuk pengguna, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Pengguna dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

   > **Note**
   >
   > Jika kebijakan yang Anda pilih adalah kebijakan inline yang dilampirkan langsung ke pengguna, maka tabel ringkasan layanan muncul. Jika kebijakan tersebut merupakan kebijakan sebaris yang dilampirkan dari grup, maka Anda akan dibawa ke dokumen JSON kebijakan untuk grup tersebut. Jika kebijakan tersebut adalah kebijakan yang dikelola, maka Anda dibawa ke ringkasan layanan untuk kebijakan tersebut di halaman Kebijakan.
7. Di daftar tindakan ringkasan layanan, pilih nama tindakan yang ingin Anda lihat.

Melihat ringkasan tindakan untuk kebijakan yang dilampirkan pada peran

Anda dapat melihat ringkasan tindakan untuk setiap kebijakan yang dilampirkan ke peran.

Untuk melihat ringkasan tindakan untuk kebijakan yang terlampir pada peran

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Dalam daftar peran, pilih nama peran yang kebijakannya ingin Anda lihat.
4. Di Ringkasan untuk peran, lihat Izin untuk melihat daftar kebijakan yang terlampir pada peran tersebut.
5. Dalam tabel kebijakan untuk peran tersebut, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Peran dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.
7. Di daftar tindakan ringkasan layanan, pilih nama tindakan yang ingin Anda lihat.

Contoh ringkasan kebijakan

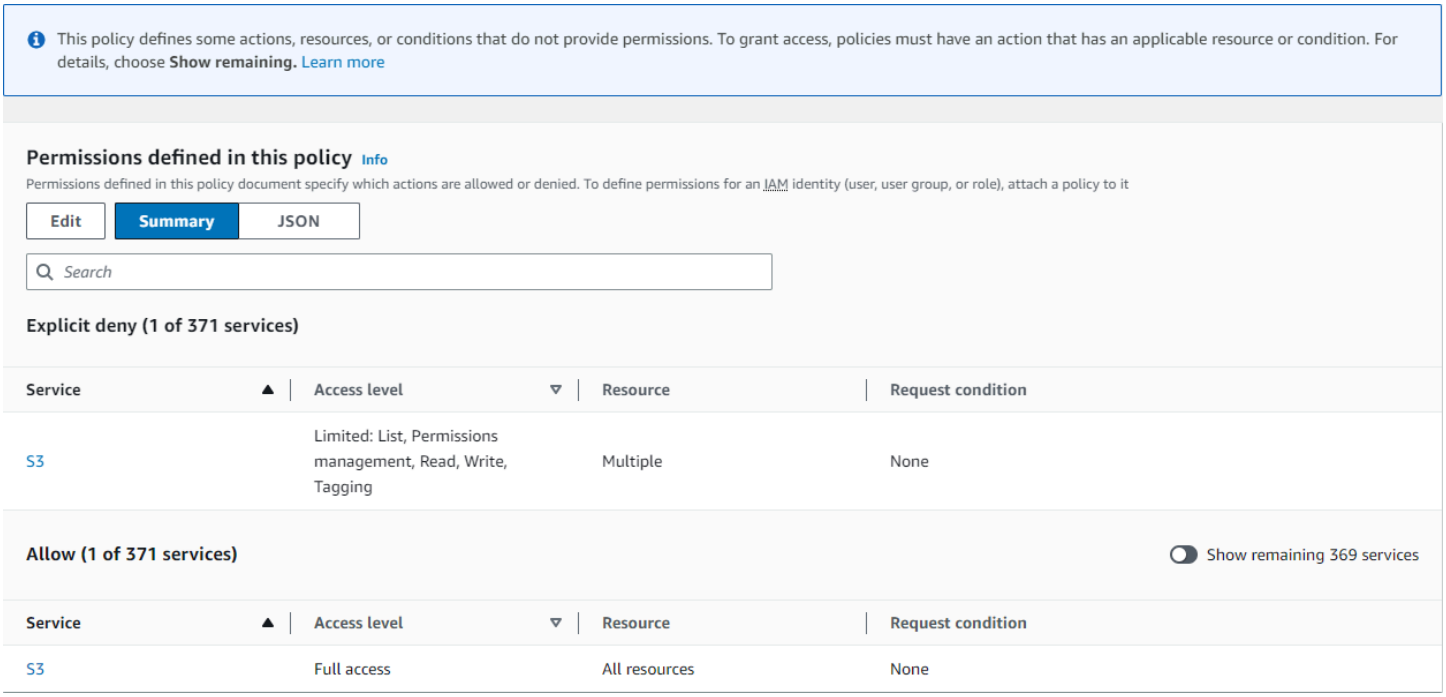
Contoh berikut mencakup kebijakan JSON dengan ringkasan kebijakan terkait, ringkasan layanan, dan ringkasan tindakan untuk membantu Anda memahami izin yang diberikan melalui kebijakan.

Kebijakan 1: DenyCustomerBucket

Kebijakan ini menunjukkan izin dan penolakan untuk layanan yang sama.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
            "Resource": ["*"]
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

DenyCustomerBucketRingkasan Kebijakan:



DenyCustomerBucket Ringkasan Layanan S3 (penolakan eksplisit):

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

GetObject (Baca) Ringkasan Tindakan:

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

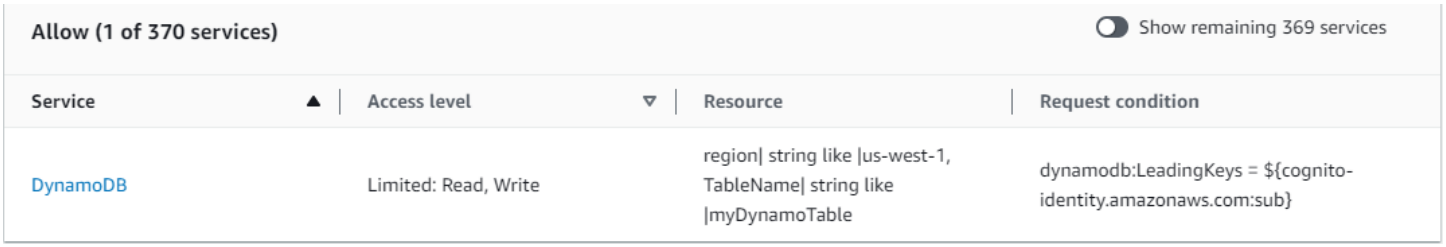
Kebijakan 2: DynamoDbRowCognito ID

Kebijakan ini menyediakan akses tingkat baris ke Amazon DynamoDB berdasarkan ID Amazon Cognito pengguna.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

DynamoDbRowCognitoRingkasan Kebijakan ID:



DynamoDbRowCognitoID DynamoDB (Izinkan) Ringkasan Layanan:

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

GetItem (Daftar) Ringkasan Tindakan:

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

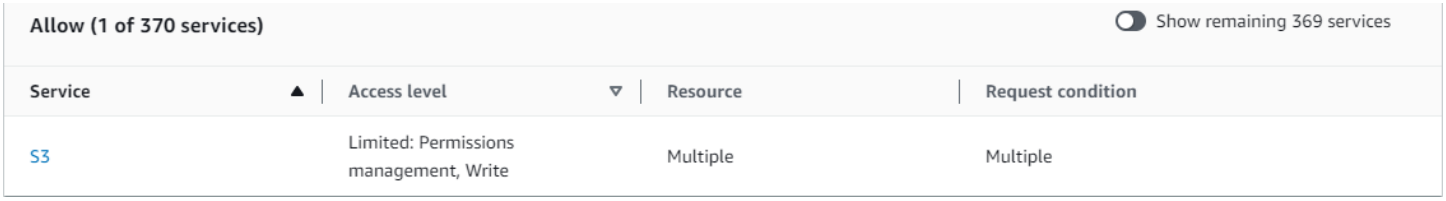
Kebijakan 3: MultipleResourceCondition

Kebijakan ini mencakup beberapa sumber daya dan ketentuan.

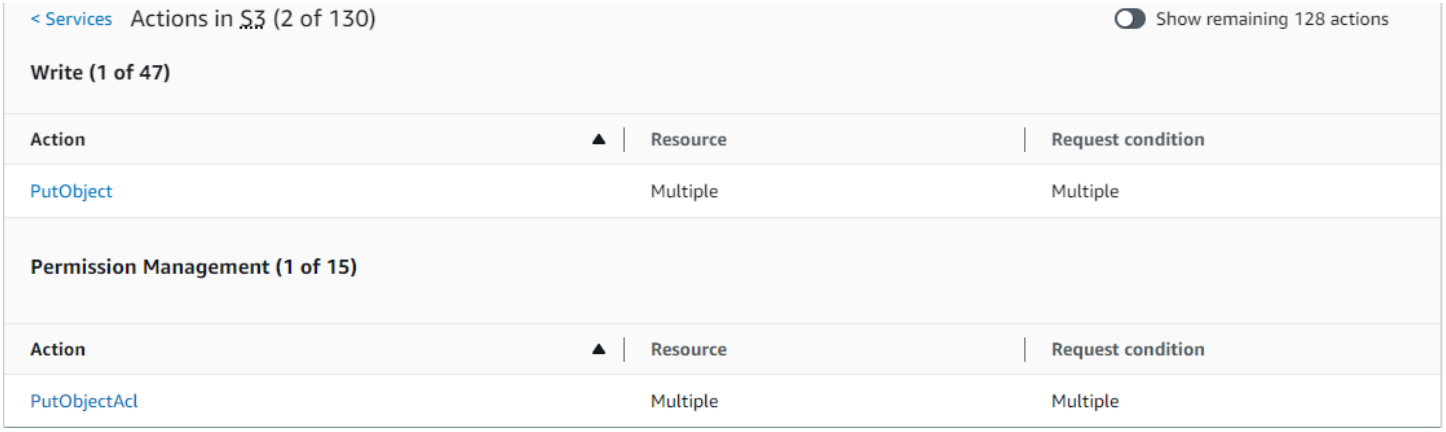
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

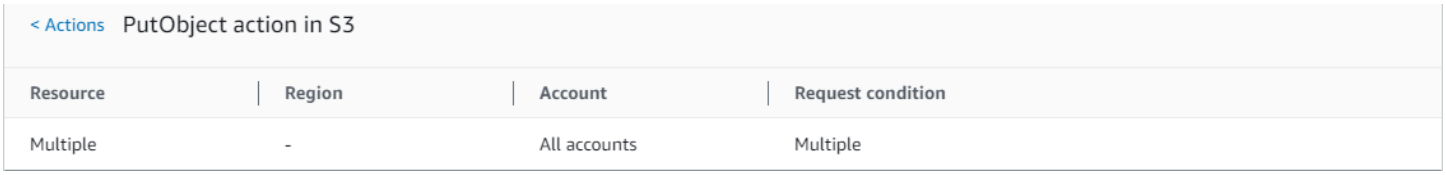
MultipleResourceConditionRingkasan Kebijakan:



MultipleResourceCondition Ringkasan Layanan S3 (Izinkan):



PutObject (Tulis) Ringkasan Tindakan:



Kebijakan 4: EC2_troubleshoot

Kebijakan berikut memungkinkan pengguna untuk mendapatkan tangkapan layar instans Amazon EC2 yang dapat membantu mengatasi masalah EC2. Kebijakan ini juga mengizinkan melihat informasi tentang item di bucket pengembang Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

EC2_Troubleshoot Ringkasan Kebijakan:

Allow (2 of 370 services) — Show remaining 368 services

Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName| string like |developer	None

EC2_Troubleshoot S3 (Izinkan) Ringkasan Layanan:

< Services Actions in S3 (1 of 130) Show remaining 129 actions

List (1 of 7)

Action	Resource	Request condition
ListBucket	BucketName| string like |developer	None

ListBucket (Daftar) Ringkasan Tindakan:

< Actions ListBucket action in S3

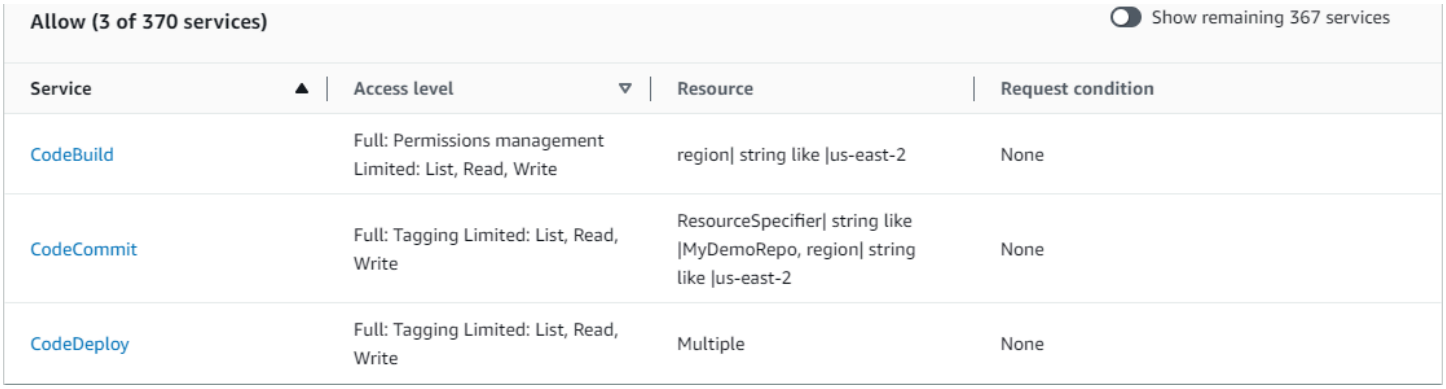
Resource	Region	Account	Request condition
BucketName| string like |developer	-	All accounts	None

Kebijakan 5: CodeBuild _ CodeCommit _ CodeDeploy

Kebijakan ini menyediakan akses ke sumber daya tertentu CodeBuild CodeCommit, dan CodeDeploy sumber daya. Karena sumber daya ini khusus untuk setiap layanan, mereka hanya muncul dengan layanan yang cocok. Jika Anda menyertakan sumber daya yang tidak cocok dengan layanan apa pun dalam elemen `Action` tersebut, sumber daya akan muncul di semua ringkasan tindakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
            ]
        }
    ]
}
```

CodeBuild_ CodeCommit _ Ringkasan CodeDeploy Kebijakan:



Allow (3 of 370 services) Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

CodeBuild_ CodeCommit _ CodeDeploy CodeBuild (Izinkan) Ringkasan Layanan:

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

CodeBuild_ CodeCommit _ CodeDeploy StartBuild (Tulis) Ringkasan Tindakan:

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

Izin diperlukan untuk mengakses sumber daya IAM

Sumber Daya adalah objek dalam suatu layanan. IAMsumber daya termasuk grup, pengguna, peran, dan kebijakan. Jika Anda masuk dengan Pengguna root akun AWS kredensyal, Anda tidak memiliki batasan dalam mengelola IAM kredensyal atau sumber daya. IAM Namun, IAM pengguna harus secara eksplisit diberi izin untuk mengelola kredensyal atau sumber daya. IAM Anda dapat melakukannya dengan melampirkan kebijakan berbasis identitas kepada pengguna.

> **Note**
>
> Sepanjang AWS dokumentasi, ketika kami merujuk pada IAM kebijakan tanpa menyebutkan kategori tertentu, yang kami maksud adalah kebijakan berbasis identitas yang dikelola pelanggan. Untuk perincian tentang kategori kebijakan, lihat the section called “Kebijakan dan Izin”.

Izin untuk mengurus identitas IAM

Izin yang diperlukan untuk mengelola IAM grup, pengguna, peran, dan kredensyal biasanya sesuai dengan API tindakan untuk tugas tersebut. Misalnya, untuk membuat IAM pengguna, Anda harus memiliki `iam:CreateUser` izin yang memiliki API perintah yang sesuai: `CreateUser`. Untuk mengizinkan IAM pengguna membuat IAM pengguna lain, Anda dapat melampirkan IAM kebijakan seperti berikut ini kepada pengguna tersebut:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Dalam kebijakan, nilai elemen `Resource` bergantung pada tindakan dan sumber daya yang dapat dipengaruhi oleh tindakan tersebut. Dalam contoh sebelumnya, kebijakan ini memungkinkan pengguna untuk membuat pengguna mana pun (* adalah wildcard yang cocok dengan semua string). Sebaliknya, kebijakan yang memungkinkan pengguna untuk mengubah hanya kunci akses mereka sendiri (APItindakan `CreateAccessKey`dan `UpdateAccessKey`) biasanya memiliki

`Resource` elemen. Dalam hal ini ARN termasuk variabel (`${aws:username}`) yang menyelesaikan nama pengguna saat ini, seperti dalam contoh berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Dalam contoh sebelumnya, `${aws:username}` adalah variabel yang menyelesaikan nama pengguna saat ini. Untuk informasi lebih lanjut tentang variabel-variable kebijakan, lihat Elemen kebijakan IAM: Variabel dan tanda.

Menggunakan karakter wildcard (*) dalam nama tindakan sering kali mempermudah pemberian izin untuk semua tindakan yang terkait dengan tugas tertentu. Misalnya, untuk memungkinkan pengguna melakukan IAM tindakan apa pun, Anda dapat menggunakannya `iam:*` untuk tindakan tersebut. Agar pengguna dapat melakukan tindakan terkait hanya untuk mengakses kunci, Anda dapat menggunakan `iam:*AccessKey*` di elemen `Action` pernyataan kebijakan. Hal ini memberi pengguna izin untuk melakukan tindakan `CreateAccessKey`, `DeleteAccessKey`, `GetAccessKeyLastUsed`, `ListAccessKeys`, dan `UpdateAccessKey`. (Jika tindakan ditambahkan ke IAM in the future yang memiliki "AccessKey" dalam nama, menggunakan `iam:*AccessKey*` untuk `Action` elemen juga akan memberikan izin pengguna untuk tindakan baru itu.) Contoh berikut menunjukkan kebijakan yang memungkinkan pengguna untuk melakukan

semua tindakan yang berkaitan dengan kunci akses mereka sendiri (ganti *account-id* dengan Akun AWS ID):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/${aws:username}"
  }
}
```

Beberapa tugas, seperti menghapus grup, melibatkan beberapa tindakan: Anda terlebih dahulu harus menghapus pengguna dari grup, kemudian memisahkan atau menghapus kebijakan grup, dan kemudian menghapusnya. Jika Anda ingin pengguna dapat menghapus grup, Anda harus yakin untuk memberikan izin kepada pengguna untuk melakukan semua tindakan terkait.

Izin untuk bekerja di AWS Management Console

Contoh sebelumnya menunjukkan kebijakan yang memungkinkan pengguna untuk melakukan tindakan dengan AWS CLIatau AWS SDKs.

Saat pengguna bekerja dengan konsol, konsol mengeluarkan permintaan IAM untuk mencantumkan grup, pengguna, peran, dan kebijakan, serta untuk mendapatkan kebijakan yang terkait dengan grup, pengguna, atau peran. Konsol juga mengeluarkan permintaan untuk mendapatkan Akun AWS informasi dan informasi tentang kepala sekolah. Prinsipalnya adalah pengguna yang mengajukan permintaan di konsol.

Secara umum, untuk melakukan tindakan, Anda hanya boleh memiliki tindakan yang sesuai yang termasuk dalam kebijakan. Untuk membuat pengguna, Anda perlu izin untuk menghubungi tindakan `CreateUser`. Sering kali, ketika Anda menggunakan konsol untuk melakukan tindakan, Anda harus memiliki izin untuk menampilkan, membuat daftar, mendapatkan, atau melihat sumber daya di konsol. Ini diperlukan agar Anda dapat menavigasi melalui konsol untuk membuat tindakan yang ditentukan. Misalnya, jika pengguna Jorge ingin menggunakan konsol untuk mengubah kunci aksesnya sendiri, ia pergi ke IAM konsol dan memilih Pengguna. Tindakan ini menyebabkan konsol untuk membuat permintaan `ListUsers`. Jika Jorge tidak memiliki izin untuk tindakan `iam:ListUsers`, konsol ditolak akses ketika mencoba untuk membuat daftar pengguna. Sebagai hasilnya, Jorge tidak bisa mendapatkan nama dan access key-nya sendiri, meskipun dia memiliki izin untuk tindakan `CreateAccessKey` dan `UpdateAccessKey`.

Jika Anda ingin memberikan izin kepada pengguna untuk mengelola grup, pengguna, peran, kebijakan, dan kredensi dengan AWS Management Console, Anda perlu menyertakan izin untuk tindakan yang dilakukan konsol. Untuk beberapa contoh kebijakan yang dapat Anda gunakan untuk memberikan izin kepada pengguna, lihat Contoh kebijakan untuk mengelola sumber daya IAM.

Berikan izin di seluruh AWS rekening

Anda dapat langsung memberikan IAM pengguna di akun Anda sendiri akses ke sumber daya Anda. Jika pengguna dari akun lain memerlukan akses ke sumber daya Anda, Anda dapat membuat IAM peran, yang merupakan entitas yang menyertakan izin tetapi tidak terkait dengan pengguna tertentu. Pengguna dari akun lain kemudian dapat menggunakan peran dan mengakses sumber daya sesuai dengan izin yang Anda tetapkan untuk peran tersebut. Untuk informasi selengkapnya, lihat Akses untuk IAM pengguna lain Akun AWS yang Anda miliki.

> **Note**
>
> Beberapa layanan mendukung kebijakan berbasis sumber daya seperti yang dijelaskan dalam Kebijakan berbasis identitas dan kebijakan berbasis sumber daya (seperti Amazon S3, Amazon, dan Amazon). SNS SQS Untuk layanan tersebut, alternatif untuk menggunakan peran adalah dengan memberikan kebijakan ke sumber daya (bucket, topik, atau antrean) yang ingin Anda bagikan. Kebijakan berbasis sumber daya dapat menentukan AWS akun yang memiliki izin untuk mengakses sumber daya.

Izin satu layanan untuk mengakses layanan lainnya

Banyak AWS layanan akses lainnya AWS layanan. Sebagai contoh, beberapa AWS layanan—termasuk EMR Amazon, Elastic Load Balancing, EC2 dan Amazon Auto EC2 Scaling—mengelola instans Amazon. Lainnya AWS layanan memanfaatkan bucket Amazon S3, topik Amazon, SQS antrian AmazonSNS, dan sebagainya.

Skenario untuk mengelola izin dalam kasus-kasus ini berbeda-beda menurut layanan. Berikut ini beberapa contoh cara izin ditangani untuk layanan yang berbeda:

- Di Amazon EC2 Auto Scaling, pengguna harus memiliki izin untuk menggunakan Auto Scaling, tetapi tidak perlu secara eksplisit diberikan izin untuk mengelola instans Amazon. EC2

- Masuk AWS Data Pipeline, IAM peran menentukan apa yang dapat dilakukan pipeline; pengguna memerlukan izin untuk mengambil peran tersebut. (Untuk detailnya, lihat Memberikan Izin ke Pipelines dengan di IAM AWS Data Pipeline Panduan Pengembang.)

Untuk detail tentang cara mengkonfigurasi izin dengan benar sehingga AWS layanan dapat menyelesaikan tugas yang Anda inginkan, merujuk ke dokumentasi untuk layanan yang Anda panggil. Untuk mempelajari cara membuat peran untuk layanan, lihat Membuat peran untuk mendelegasikan izin ke layanan AWS.

Mengkonfigurasi layanan dengan IAM peran untuk bekerja atas nama Anda

Bila Anda ingin mengkonfigurasi AWS Layanan untuk bekerja atas nama Anda, Anda biasanya menyediakan ARN untuk IAM peran yang mendefinisikan apa layanan diizinkan untuk melakukan. AWS memeriksa untuk memastikan bahwa Anda memiliki izin untuk meneruskan peran ke layanan. Untuk informasi selengkapnya, lihat Berikan izin pengguna untuk meneruskan peran ke layanan AWS.

Tindakan yang diperlukan

Tindakan adalah hal-hal yang dapat Anda lakukan ke sumber daya, seperti melihat, membuat, mengedit, dan menghapus sumber daya tersebut. Tindakan ditentukan oleh masing-masing AWS layanan.

Untuk mengizinkan seseorang melakukan suatu tindakan, Anda harus menyertakan tindakan yang diperlukan dalam kebijakan yang berlaku untuk identitas panggilan atau sumber daya yang terpengaruh. Secara umum, untuk memberikan izin yang diperlukan untuk melakukan tindakan, Anda harus menyertakan tindakan tersebut dalam kebijakan Anda. Misalnya, untuk membuat pengguna, Anda perlu menambahkan CreateUser tindakan ke kebijakan Anda.

Dalam beberapa kasus, suatu tindakan mungkin mengharuskan Anda menyertakan tindakan terkait tambahan dalam kebijakan Anda. Misalnya, untuk memberikan izin bagi seseorang untuk membuat direktori di AWS Directory Service menggunakan `ds:CreateDirectory` operasi, Anda harus memasukkan tindakan berikut dalam kebijakan mereka:

- `ds:CreateDirectory`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:CreateSecurityGroup`

- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

Saat Anda membuat atau mengedit kebijakan menggunakan editor visual, Anda menerima peringatan dan petunjuk untuk membantu Anda memilih semua tindakan yang diperlukan untuk kebijakan Anda.

Untuk informasi selengkapnya tentang izin yang diperlukan untuk membuat direktori di AWS Directory Service, lihat Contoh 2: Izinkan Pengguna Membuat Direktori.

Contoh kebijakan untuk mengelola sumber daya IAM

Berikut adalah contoh kebijakan IAM yang memungkinkan pengguna melakukan tugas yang terkait dengan pengelolaan pengguna IAM, grup, dan kredensial. Ini termasuk kebijakan yang mengizinkan pengguna mengelola kata sandi, kunci akses, dan perangkat multi-factor authentication (MFA) mereka sendiri.

Untuk contoh kebijakan yang memungkinkan pengguna melakukan tugas dengan AWS layanan lain, seperti Amazon S3, Amazon EC2, dan DynamoDB, lihat. Contoh kebijakan berbasis identitas IAM

Topik

- Memungkinkan pengguna untuk membuat daftar grup akun, pengguna, kebijakan, dan lainnya untuk tujuan pelaporan
- Memungkinkan pengguna untuk mengelola keanggotaan grup
- Izinkan pengguna untuk mengelola pengguna IAM
- Izinkan pengguna mengatur kebijakan kata sandi akun
- Memungkinkan pengguna membuat dan mengambil laporan kredensial IAM
- Izinkan semua tindakan IAM (akses admin)

Memungkinkan pengguna untuk membuat daftar grup akun, pengguna, kebijakan, dan lainnya untuk tujuan pelaporan

Kebijakan berikut memungkinkan pengguna untuk menghubungi setiap tindakan IAM yang dimulai dengan string `Get` atau `List`, dan untuk membuat laporan. Untuk melihat contoh kebijakan, lihat IAM: Mengizinkan akses hanya-baca ke konsol IAM.

Memungkinkan pengguna untuk mengelola keanggotaan grup

Kebijakan berikut memungkinkan pengguna untuk memperbarui keanggotaan grup yang dipanggil MarketingGroup. Untuk melihat contoh kebijakan, lihat IAM: Memungkinkan mengelola keanggotaan grup secara terprogram dan di konsol.

Izinkan pengguna untuk mengelola pengguna IAM

Kebijakan berikut memungkinkan pengguna untuk melakukan semua tugas yang terkait dengan mengelola pengguna IAM tetapi tidak melakukan tindakan pada entitas lain, seperti membuat grup atau kebijakan. Tindakan yang diizinkan meliputi hal berikut ini:

- Membuat pengguna (tindakan `CreateUser`).
- Menghapus pengguna. Tugas ini memerlukan izin untuk melakukan semua tindakan berikut: `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup`, dan `DeleteUser`.
- Mencantumkan pengguna dalam akun dan dalam grup (tindakan `GetUser`, `ListUsers`, dan `ListGroupsForUser`).
- Membuat daftar dan menghapus kebijakan untuk pengguna (tindakan `ListUserPolicies`, `ListAttachedUserPolicies`, `DetachUserPolicy`, `DeleteUserPolicy`)
- Mengganti nama atau mengubah jalur untuk pengguna (tindakan `UpdateUser`). Elemen `Resource` harus menyertakan ARN yang mencakup jalur sumber maupun jalur target. Untuk informasi lebih lanjut tentang jalur, lihat Nama dan jalur yang ramah.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

Sejumlah izin yang disertakan dalam kebijakan sebelumnya memungkinkan pengguna melakukan tugas dalam AWS Management Console. Pengguna yang melakukan tugas terkait pengguna dari, AWS SDK AWS CLI, atau API kueri HTTP IAM saja mungkin tidak memerlukan izin tertentu. Misalnya, jika pengguna sudah mengetahui kebijakan ARN untuk melepas dari pengguna, mereka tidak memerlukan izin `iam:ListAttachedUserPolicies`. Daftar pasti izin yang diperlukan pengguna tergantung pada tugas yang harus dilakukan pengguna saat mengelola pengguna lain.

Izin berikut dalam kebijakan ini memungkinkan akses ke tugas pengguna melalui AWS Management Console:

- `iam:GetAccount*`
- `iam:ListAccount*`

Izinkan pengguna mengatur kebijakan kata sandi akun

Anda mungkin memberikan izin kepada beberapa pengguna untuk mendapatkan dan memperbarui kebijakan kata sandi Anda Akun AWS. Untuk melihat contoh kebijakan, lihat IAM: Memungkinkan pengaturan persyaratan kata sandi akun secara terprogram dan di konsol.

Memungkinkan pengguna membuat dan mengambil laporan kredensial IAM

Anda dapat memberikan izin kepada pengguna untuk membuat dan mengunduh laporan yang mencantumkan semua pengguna di akun Anda Akun AWS. Laporan tersebut juga mencantumkan status berbagai kredensial pengguna, termasuk sandi, access key, perangkat MFA, dan sertifikat penandatanganan. Untuk informasi lebih lanjut mengenai laporan kredensial, lihat Hasilkan laporan kredensi untuk Anda Akun AWS. Untuk melihat contoh kebijakan, lihat IAM: Menghasilkan dan mengambil laporan kredensi IAM.

Izinkan semua tindakan IAM (akses admin)

Anda dapat memberikan izin administratif kepada beberapa pengguna untuk melakukan semua tindakan di IAM, termasuk mengelola kata sandi, kunci akses, perangkat MFA, dan sertifikat pengguna. Kebijakan contoh berikut memberikan izin ini.

> **Warning**
>
> Jika Anda memberikan akses penuh kepada pengguna ke IAM, tidak ada batasan untuk izin yang dapat diberikan pengguna kepadanya atau orang lain. Pengguna dapat membuat entitas IAM baru (pengguna atau peran) dan memberikan entitas tersebut akses penuh ke semua sumber daya di Anda Akun AWS. Ketika Anda memberi pengguna akses penuh ke IAM, Anda secara efektif memberi mereka akses penuh ke semua sumber daya di Anda Akun AWS. Ini termasuk akses untuk menghapus semua sumber daya. Anda harus memberikan izin ini hanya kepada administrator tepercaya, dan Anda harus memberlakukan Autentikasi Multi-Faktor (MFA) untuk administrator ini.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

Contoh kode untuk IAM menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan IAM kit pengembangan AWS perangkat lunak (SDK).

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Contoh kode

- Gunakan CreateServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan CreateUser dengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevice dengan CLI
- Gunakan DeactivateMfaDevice dengan CLI
- Gunakan DeleteAccessKey dengan AWS SDK atau CLI
- Gunakan DeleteAccountAlias dengan AWS SDK atau CLI
- Gunakan DeleteAccountPasswordPolicy dengan CLI
- Gunakan DeleteGroup dengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicy dengan AWS SDK atau CLI
- Gunakan DeleteInstanceProfile dengan AWS SDK atau CLI
- Gunakan DeleteLoginProfile dengan CLI
- Gunakan DeleteOpenIdConnectProvider dengan CLI
- Gunakan DeletePolicy dengan AWS SDK atau CLI
- Gunakan DeletePolicyVersion dengan a CLI
- Gunakan DeleteRole dengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundary dengan CLI
- Gunakan DeleteRolePolicy dengan AWS SDK atau CLI
- Gunakan DeleteSAMLProvider dengan AWS SDK atau CLI
- Gunakan DeleteServerCertificate dengan AWS SDK atau CLI
- Gunakan DeleteServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan DeleteSigningCertificate dengan CLI
- Gunakan DeleteUser dengan AWS SDK atau CLI
- Gunakan DeleteUserPermissionsBoundary dengan CLI
- Gunakan DeleteUserPolicy dengan AWS SDK atau CLI
- Gunakan DeleteVirtualMfaDevice dengan CLI
- Gunakan DetachGroupPolicy dengan CLI
- Gunakan DetachRolePolicy dengan AWS SDK atau CLI
- Gunakan DetachUserPolicy dengan AWS SDK atau CLI
- Gunakan EnableMfaDevice dengan CLI

- Gunakan GenerateCredentialReport dengan AWS SDK atau CLI

- Gunakan GenerateServiceLastAccessedDetails dengan CLI
- Gunakan GetAccessKeyLastUsed dengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetails dengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicy dengan AWS SDK atau CLI
- Gunakan GetAccountSummary dengan AWS SDK atau CLI
- Gunakan GetContextKeysForCustomPolicy dengan CLI
- Gunakan GetContextKeysForPrincipalPolicy dengan CLI
- Gunakan GetCredentialReport dengan AWS SDK atau CLI
- Gunakan GetGroup dengan CLI
- Gunakan GetGroupPolicy dengan CLI
- Gunakan GetInstanceProfile dengan CLI
- Gunakan GetLoginProfile dengan a CLI
- Gunakan GetOpenIdConnectProvider dengan CLI
- Gunakan GetPolicy dengan AWS SDK atau CLI
- Gunakan GetPolicyVersion dengan AWS SDK atau CLI
- Gunakan GetRole dengan AWS SDK atau CLI
- Gunakan GetRolePolicy dengan CLI
- Gunakan GetSamlProvider dengan CLI
- Gunakan GetServerCertificate dengan AWS SDK atau CLI
- Gunakan GetServiceLastAccessedDetails dengan CLI
- Gunakan GetServiceLastAccessedDetailsWithEntities dengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatus dengan AWS SDK atau CLI
- Gunakan GetUser dengan AWS SDK atau CLI
- Gunakan GetUserPolicy dengan a CLI
- Gunakan ListAccessKeys dengan AWS SDK atau CLI
- Gunakan ListAccountAliases dengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPolicies dengan a CLI
- Gunakan ListAttachedRolePolicies dengan AWS SDK atau CLI
- Gunakan ListAttachedUserPolicies dengan CLI
- Gunakan ListEntitiesForPolicy dengan a CLI

- Gunakan ListGroupPolicies dengan CLI
- Gunakan ListGroups dengan AWS SDK atau CLI
- Gunakan ListGroupsForUser dengan CLI
- Gunakan ListInstanceProfiles dengan CLI
- Gunakan ListInstanceProfilesForRole dengan CLI
- Gunakan ListMfaDevices dengan CLI
- Gunakan ListOpenIdConnectProviders dengan CLI
- Gunakan ListPolicies dengan AWS SDK atau CLI
- Gunakan ListPolicyVersions dengan CLI
- Gunakan ListRolePolicies dengan AWS SDK atau CLI
- Gunakan ListRoleTags dengan CLI
- Gunakan ListRoles dengan AWS SDK atau CLI
- Gunakan ListSAMLProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificates dengan AWS SDK atau CLI
- Gunakan ListSigningCertificates dengan CLI
- Gunakan ListUserPolicies dengan AWS SDK atau CLI
- Gunakan ListUserTags dengan CLI
- Gunakan ListUsers dengan AWS SDK atau CLI
- Gunakan ListVirtualMfaDevices dengan a CLI
- Gunakan PutGroupPolicy dengan AWS SDK atau CLI
- Gunakan PutRolePermissionsBoundary dengan CLI
- Gunakan PutRolePolicy dengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundary dengan CLI
- Gunakan PutUserPolicy dengan AWS SDK atau CLI
- Gunakan RemoveClientIdFromOpenIdConnectProvider dengan CLI
- Gunakan RemoveRoleFromInstanceProfile dengan CLI
- Gunakan RemoveUserFromGroup dengan AWS SDK atau CLI
- Gunakan ResyncMfaDevice dengan CLI
- Gunakan SetDefaultPolicyVersion dengan CLI
- Gunakan TagRole dengan CLI

Contoh kode untuk IAM menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan IAM kit pengembangan AWS perangkat lunak (SDK).

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain Layanan AWS.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Memulai

Halo IAM

Contoh kode berikut menunjukkan cara untuk mulai menggunakanIAM.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for .NET APIReferensi.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Kode untuk CMakeLists file.txtCMake.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Kode untuk file sumber iam.cpp.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
<<
                          std::endl;
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
```

```
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for C++ APIReferensi.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
```

```
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
```

```
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      for (const policy of page.Policies) {
        console.log(`${policy.PolicyName}`);
        policyCount++;
      }
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for JavaScript APIReferensi.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import boto3


def main():
    """
    Lists the managed policies in your AWS account using the AWS SDK for Python
 (Boto3).
    """
    iam = boto3.client("iam")

    try:
        # Get a paginator for the list_policies operation
        paginator = iam.get_paginator("list_policies")

        # Iterate through the pages of results
        for page in paginator.paginate(Scope="All", OnlyAttached=False):
            for policy in page["Policies"]:
                print(f"Policy name: {policy['PolicyName']}")
                print(f"  Policy ARN: {policy['Arn']}")
    except boto3.exceptions.BotoCoreError as e:
        print(f"Encountered an error while listing policies: {e}")


if __name__ == "__main__":
    main()
```

- Untuk API detailnya, lihat ListPolicies AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
require 'aws-sdk-iam'
require 'logger'

# IAMManager is a class responsible for managing IAM operations
# such as listing all IAM policies in the current AWS account.
class IAMManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all IAM policies in the current AWS account.
  def list_policies
    @logger.info('Here are the IAM policies in your account:')

    paginator = @client.list_policies
    policies = []

    paginator.each_page do |page|
      policies.concat(page.policies)
    end

    if policies.empty?
      @logger.info("You don't have any IAM policies.")
    else
      policies.each do |policy|
        @logger.info("- #{policy.policy_name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  iam_client = Aws::IAM::Client.new
  manager = IAMManager.new(iam_client)
  manager.list_policies
end
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dari src/bin/hello .rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), SdkError<ListPoliciesError>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

Dari src/ .rsiam-service-lib.

```
pub async fn list_policies(
```

```
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Untuk API detailnya, lihat ListPolicies AWSSDKuntuk API referensi Rust.

Contoh kode

- Contoh dasar untuk IAM menggunakan AWS SDKs
  - Halo IAM
  - Pelajari dasar-dasar IAM dengan AWS SDK
  - Tindakan untuk IAM menggunakan AWS SDKs
    - Gunakan AddClientIdToOpenIdConnectProvider dengan CLI
    - Gunakan AddRoleToInstanceProfile dengan CLI
    - Gunakan AddUserToGroup dengan AWS SDK atau CLI
    - Gunakan AttachGroupPolicy dengan CLI

- Gunakan AttachRolePolicy dengan AWS SDK atau CLI
- Gunakan AttachUserPolicy dengan AWS SDK atau CLI
- Gunakan ChangePassword dengan CLI
- Gunakan CreateAccessKey dengan AWS SDK atau CLI
- Gunakan CreateAccountAlias dengan AWS SDK atau CLI
- Gunakan CreateGroup dengan AWS SDK atau CLI
- Gunakan CreateInstanceProfile dengan AWS SDK atau CLI
- Gunakan CreateLoginProfile dengan CLI
- Gunakan CreateOpenIdConnectProvider dengan a CLI
- Gunakan CreatePolicy dengan AWS SDK atau CLI
- Gunakan CreatePolicyVersion dengan AWS SDK atau CLI
- Gunakan CreateRole dengan AWS SDK atau CLI
- Gunakan CreateSAMLProvider dengan AWS SDK atau CLI
- Gunakan CreateServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan CreateUser dengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevice dengan CLI
- Gunakan DeactivateMfaDevice dengan CLI
- Gunakan DeleteAccessKey dengan AWS SDK atau CLI
- Gunakan DeleteAccountAlias dengan AWS SDK atau CLI
- Gunakan DeleteAccountPasswordPolicy dengan CLI
- Gunakan DeleteGroup dengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicy dengan AWS SDK atau CLI
- Gunakan DeleteInstanceProfile dengan AWS SDK atau CLI
- Gunakan DeleteLoginProfile dengan CLI
- Gunakan DeleteOpenIdConnectProvider dengan CLI
- Gunakan DeletePolicy dengan AWS SDK atau CLI
- Gunakan DeletePolicyVersion dengan a CLI
- Gunakan DeleteRole dengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundary dengan CLI
- Gunakan DeleteRolePolicy dengan AWS SDK atau CLI

- Gunakan DeleteSAMLProvider dengan AWS SDK atau CLI
- Gunakan DeleteServerCertificate dengan AWS SDK atau CLI
- Gunakan DeleteServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan DeleteSigningCertificate dengan CLI
- Gunakan DeleteUser dengan AWS SDK atau CLI
- Gunakan DeleteUserPermissionsBoundary dengan CLI
- Gunakan DeleteUserPolicy dengan AWS SDK atau CLI
- Gunakan DeleteVirtualMfaDevice dengan CLI
- Gunakan DetachGroupPolicy dengan CLI
- Gunakan DetachRolePolicy dengan AWS SDK atau CLI
- Gunakan DetachUserPolicy dengan AWS SDK atau CLI
- Gunakan EnableMfaDevice dengan CLI
- Gunakan GenerateCredentialReport dengan AWS SDK atau CLI
- Gunakan GenerateServiceLastAccessedDetails dengan CLI
- Gunakan GetAccessKeyLastUsed dengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetails dengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicy dengan AWS SDK atau CLI
- Gunakan GetAccountSummary dengan AWS SDK atau CLI
- Gunakan GetContextKeysForCustomPolicy dengan CLI
- Gunakan GetContextKeysForPrincipalPolicy dengan CLI
- Gunakan GetCredentialReport dengan AWS SDK atau CLI
- Gunakan GetGroup dengan CLI
- Gunakan GetGroupPolicy dengan CLI
- Gunakan GetInstanceProfile dengan CLI
- Gunakan GetLoginProfile dengan a CLI
- Gunakan GetOpenIdConnectProvider dengan CLI
- Gunakan GetPolicy dengan AWS SDK atau CLI
- Gunakan GetPolicyVersion dengan AWS SDK atau CLI
- Gunakan GetRole dengan AWS SDK atau CLI
- Gunakan GetRolePolicy dengan CLI

- Gunakan GetSamlProvider dengan CLI
- Gunakan GetServerCertificate dengan AWS SDK atau CLI
- Gunakan GetServiceLastAccessedDetails dengan CLI
- Gunakan GetServiceLastAccessedDetailsWithEntities dengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatus dengan AWS SDK atau CLI
- Gunakan GetUser dengan AWS SDK atau CLI
- Gunakan GetUserPolicy dengan a CLI
- Gunakan ListAccessKeys dengan AWS SDK atau CLI
- Gunakan ListAccountAliases dengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPolicies dengan a CLI
- Gunakan ListAttachedRolePolicies dengan AWS SDK atau CLI
- Gunakan ListAttachedUserPolicies dengan CLI
- Gunakan ListEntitiesForPolicy dengan a CLI
- Gunakan ListGroupPolicies dengan CLI
- Gunakan ListGroups dengan AWS SDK atau CLI
- Gunakan ListGroupsForUser dengan CLI
- Gunakan ListInstanceProfiles dengan CLI
- Gunakan ListInstanceProfilesForRole dengan CLI
- Gunakan ListMfaDevices dengan CLI
- Gunakan ListOpenIdConnectProviders dengan CLI
- Gunakan ListPolicies dengan AWS SDK atau CLI
- Gunakan ListPolicyVersions dengan CLI
- Gunakan ListRolePolicies dengan AWS SDK atau CLI
- Gunakan ListRoleTags dengan CLI
- Gunakan ListRoles dengan AWS SDK atau CLI
- Gunakan ListSAMLProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificates dengan AWS SDK atau CLI
- Gunakan ListSigningCertificates dengan CLI
- Gunakan ListUserPolicies dengan AWS SDK atau CLI
- Gunakan ListUserTags dengan CLI

- Gunakan ListUsers dengan AWS SDK atau CLI
- Gunakan ListVirtualMfaDevices dengan a CLI
- Gunakan PutGroupPolicy dengan AWS SDK atau CLI
- Gunakan PutRolePermissionsBoundary dengan CLI
- Gunakan PutRolePolicy dengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundary dengan CLI
- Gunakan PutUserPolicy dengan AWS SDK atau CLI
- Gunakan RemoveClientIdFromOpenIdConnectProvider dengan CLI
- Gunakan RemoveRoleFromInstanceProfile dengan CLI
- Gunakan RemoveUserFromGroup dengan AWS SDK atau CLI
- Gunakan ResyncMfaDevice dengan CLI
- Gunakan SetDefaultPolicyVersion dengan CLI
- Gunakan TagRole dengan CLI
- Gunakan TagUser dengan CLI
- Gunakan UntagRole dengan CLI
- Gunakan UntagUser dengan CLI
- Gunakan UpdateAccessKey dengan AWS SDK atau CLI
- Gunakan UpdateAccountPasswordPolicy dengan CLI
- Gunakan UpdateAssumeRolePolicy dengan CLI
- Gunakan UpdateGroup dengan CLI
- Gunakan UpdateLoginProfile dengan CLI
- Gunakan UpdateOpenIdConnectProviderThumbprint dengan CLI
- Gunakan UpdateRole dengan CLI
- Gunakan UpdateRoleDescription dengan CLI
- Gunakan UpdateSamlProvider dengan CLI
- Gunakan UpdateServerCertificate dengan AWS SDK atau CLI
- Gunakan UpdateSigningCertificate dengan CLI
- Gunakan UpdateUser dengan AWS SDK atau CLI
- Gunakan UploadServerCertificate dengan AWS SDK atau CLI
- Gunakan UploadSigningCertificate dengan CLI

- Skenario untuk IAM menggunakan AWS SDKs
  - Membangun dan mengelola layanan tangguh menggunakan AWS SDK
  - Buat IAM grup dan tambahkan pengguna ke grup menggunakan AWS SDK
  - Buat pengguna read-only dan read-write menggunakan IAM AWS SDK
  - Mengelola kunci IAM akses menggunakan AWS SDK
  - Mengelola IAM kebijakan menggunakan AWS SDK
  - Mengelola IAM peran menggunakan AWS SDK
  - Mengelola IAM akun Anda menggunakan AWS SDK
  - Kembalikan versi IAM kebijakan menggunakan AWS SDK
  - Bekerja dengan Pembuat IAM Kebijakan API menggunakan AWS SDK

Contoh dasar untuk IAM menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar AWS Identity and Access Management dengan AWS SDKs.

Contoh

- Halo IAM
- Pelajari dasar-dasar IAM dengan AWS SDK
- Tindakan untuk IAM menggunakan AWS SDKs
  - Gunakan AddClientIdToOpenIdConnectProvider dengan CLI
  - Gunakan AddRoleToInstanceProfile dengan CLI
  - Gunakan AddUserToGroup dengan AWS SDK atau CLI
  - Gunakan AttachGroupPolicy dengan CLI
  - Gunakan AttachRolePolicy dengan AWS SDK atau CLI
  - Gunakan AttachUserPolicy dengan AWS SDK atau CLI
  - Gunakan ChangePassword dengan CLI
  - Gunakan CreateAccessKey dengan AWS SDK atau CLI
  - Gunakan CreateAccountAlias dengan AWS SDK atau CLI
  - Gunakan CreateGroup dengan AWS SDK atau CLI
  - Gunakan CreateInstanceProfile dengan AWS SDK atau CLI
  - Gunakan CreateLoginProfile dengan CLI

- Gunakan CreateOpenIdConnectProvider dengan a CLI
- Gunakan CreatePolicy dengan AWS SDK atau CLI
- Gunakan CreatePolicyVersion dengan AWS SDK atau CLI
- Gunakan CreateRole dengan AWS SDK atau CLI
- Gunakan CreateSAMLProvider dengan AWS SDK atau CLI
- Gunakan CreateServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan CreateUser dengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevice dengan CLI
- Gunakan DeactivateMfaDevice dengan CLI
- Gunakan DeleteAccessKey dengan AWS SDK atau CLI
- Gunakan DeleteAccountAlias dengan AWS SDK atau CLI
- Gunakan DeleteAccountPasswordPolicy dengan CLI
- Gunakan DeleteGroup dengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicy dengan AWS SDK atau CLI
- Gunakan DeleteInstanceProfile dengan AWS SDK atau CLI
- Gunakan DeleteLoginProfile dengan CLI
- Gunakan DeleteOpenIdConnectProvider dengan CLI
- Gunakan DeletePolicy dengan AWS SDK atau CLI
- Gunakan DeletePolicyVersion dengan a CLI
- Gunakan DeleteRole dengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundary dengan CLI
- Gunakan DeleteRolePolicy dengan AWS SDK atau CLI
- Gunakan DeleteSAMLProvider dengan AWS SDK atau CLI
- Gunakan DeleteServerCertificate dengan AWS SDK atau CLI
- Gunakan DeleteServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan DeleteSigningCertificate dengan CLI
- Gunakan DeleteUser dengan AWS SDK atau CLI
- Gunakan DeleteUserPermissionsBoundary dengan CLI
- Gunakan DeleteUserPolicy dengan AWS SDK atau CLI

- Gunakan DeleteVirtualMfaDevice dengan CLI

- Gunakan DetachGroupPolicy dengan CLI
- Gunakan DetachRolePolicy dengan AWS SDK atau CLI
- Gunakan DetachUserPolicy dengan AWS SDK atau CLI
- Gunakan EnableMfaDevice dengan CLI
- Gunakan GenerateCredentialReport dengan AWS SDK atau CLI
- Gunakan GenerateServiceLastAccessedDetails dengan CLI
- Gunakan GetAccessKeyLastUsed dengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetails dengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicy dengan AWS SDK atau CLI
- Gunakan GetAccountSummary dengan AWS SDK atau CLI
- Gunakan GetContextKeysForCustomPolicy dengan CLI
- Gunakan GetContextKeysForPrincipalPolicy dengan CLI
- Gunakan GetCredentialReport dengan AWS SDK atau CLI
- Gunakan GetGroup dengan CLI
- Gunakan GetGroupPolicy dengan CLI
- Gunakan GetInstanceProfile dengan CLI
- Gunakan GetLoginProfile dengan a CLI
- Gunakan GetOpenIdConnectProvider dengan CLI
- Gunakan GetPolicy dengan AWS SDK atau CLI
- Gunakan GetPolicyVersion dengan AWS SDK atau CLI
- Gunakan GetRole dengan AWS SDK atau CLI
- Gunakan GetRolePolicy dengan CLI
- Gunakan GetSamlProvider dengan CLI
- Gunakan GetServerCertificate dengan AWS SDK atau CLI
- Gunakan GetServiceLastAccessedDetails dengan CLI
- Gunakan GetServiceLastAccessedDetailsWithEntities dengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatus dengan AWS SDK atau CLI
- Gunakan GetUser dengan AWS SDK atau CLI
- Gunakan GetUserPolicy dengan a CLI
- Gunakan ListAccessKeys dengan AWS SDK atau CLI

- Gunakan ListAccountAliases dengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPolicies dengan a CLI
- Gunakan ListAttachedRolePolicies dengan AWS SDK atau CLI
- Gunakan ListAttachedUserPolicies dengan CLI
- Gunakan ListEntitiesForPolicy dengan a CLI
- Gunakan ListGroupPolicies dengan CLI
- Gunakan ListGroups dengan AWS SDK atau CLI
- Gunakan ListGroupsForUser dengan CLI
- Gunakan ListInstanceProfiles dengan CLI
- Gunakan ListInstanceProfilesForRole dengan CLI
- Gunakan ListMfaDevices dengan CLI
- Gunakan ListOpenIdConnectProviders dengan CLI
- Gunakan ListPolicies dengan AWS SDK atau CLI
- Gunakan ListPolicyVersions dengan CLI
- Gunakan ListRolePolicies dengan AWS SDK atau CLI
- Gunakan ListRoleTags dengan CLI
- Gunakan ListRoles dengan AWS SDK atau CLI
- Gunakan ListSAMLProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificates dengan AWS SDK atau CLI
- Gunakan ListSigningCertificates dengan CLI
- Gunakan ListUserPolicies dengan AWS SDK atau CLI
- Gunakan ListUserTags dengan CLI
- Gunakan ListUsers dengan AWS SDK atau CLI
- Gunakan ListVirtualMfaDevices dengan a CLI
- Gunakan PutGroupPolicy dengan AWS SDK atau CLI
- Gunakan PutRolePermissionsBoundary dengan CLI
- Gunakan PutRolePolicy dengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundary dengan CLI
- Gunakan PutUserPolicy dengan AWS SDK atau CLI

- Gunakan RemoveClientIdFromOpenIdConnectProvider dengan CLI

- Gunakan RemoveRoleFromInstanceProfile dengan CLI
- Gunakan RemoveUserFromGroup dengan AWS SDK atau CLI
- Gunakan ResyncMfaDevice dengan CLI
- Gunakan SetDefaultPolicyVersion dengan CLI
- Gunakan TagRole dengan CLI
- Gunakan TagUser dengan CLI
- Gunakan UntagRole dengan CLI
- Gunakan UntagUser dengan CLI
- Gunakan UpdateAccessKey dengan AWS SDK atau CLI
- Gunakan UpdateAccountPasswordPolicy dengan CLI
- Gunakan UpdateAssumeRolePolicy dengan CLI
- Gunakan UpdateGroup dengan CLI
- Gunakan UpdateLoginProfile dengan CLI
- Gunakan UpdateOpenIdConnectProviderThumbprint dengan CLI
- Gunakan UpdateRole dengan CLI
- Gunakan UpdateRoleDescription dengan CLI
- Gunakan UpdateSamlProvider dengan CLI
- Gunakan UpdateServerCertificate dengan AWS SDK atau CLI
- Gunakan UpdateSigningCertificate dengan CLI
- Gunakan UpdateUser dengan AWS SDK atau CLI
- Gunakan UploadServerCertificate dengan AWS SDK atau CLI
- Gunakan UploadSigningCertificate dengan CLI

Halo IAM

Contoh kode berikut menunjukkan cara untuk mulai menggunakanIAM.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for .NET APIReferensi.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Kode untuk CMakeLists file.txtCMake.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Kode untuk file sumber iam.cpp.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
<<
                          std::endl;
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
```

```
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for C++ APIReferensi.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
```

```
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
```

```
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      for (const policy of page.Policies) {
        console.log(`${policy.PolicyName}`);
        policyCount++;
      }
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for JavaScript APIReferensi.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import boto3


def main():
    """
    Lists the managed policies in your AWS account using the AWS SDK for Python
 (Boto3).
    """
    iam = boto3.client("iam")

    try:
        # Get a paginator for the list_policies operation
        paginator = iam.get_paginator("list_policies")

        # Iterate through the pages of results
        for page in paginator.paginate(Scope="All", OnlyAttached=False):
            for policy in page["Policies"]:
                print(f"Policy name: {policy['PolicyName']}")
                print(f"  Policy ARN: {policy['Arn']}")
    except boto3.exceptions.BotoCoreError as e:
        print(f"Encountered an error while listing policies: {e}")


if __name__ == "__main__":
    main()
```

- Untuk API detailnya, lihat ListPolicies AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
require 'aws-sdk-iam'
require 'logger'

# IAMManager is a class responsible for managing IAM operations
# such as listing all IAM policies in the current AWS account.
class IAMManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all IAM policies in the current AWS account.
  def list_policies
    @logger.info('Here are the IAM policies in your account:')

    paginator = @client.list_policies
    policies = []

    paginator.each_page do |page|
      policies.concat(page.policies)
    end

    if policies.empty?
      @logger.info("You don't have any IAM policies.")
    else
      policies.each do |policy|
        @logger.info("- #{policy.policy_name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  iam_client = Aws::IAM::Client.new
  manager = IAMManager.new(iam_client)
  manager.list_policies
end
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dari src/bin/hello .rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), SdkError<ListPoliciesError>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

Dari src/ .rsiam-service-lib.

```
pub async fn list_policies(
```

```
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Untuk API detailnya, lihat ListPolicies AWSSDKuntuk API referensi Rust.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Pelajari dasar-dasar IAM dengan AWS SDK

Contoh kode berikut menunjukkan cara membuat pengguna dan mengambil peran.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan IAM pengguna untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat pengguna tanpa izin.
- Buat peran yang memberikan izin untuk mencantumkan bucket Amazon S3 untuk akun tersebut.
- Tambahkan kebijakan agar pengguna dapat mengambil peran tersebut.
- Asumsikan peran dan daftar bucket S3 menggunakan kredensyal sementara, lalu bersihkan sumber daya.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
```

```
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
```

```
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
```

```
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
```

```
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
```

```
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
```

```
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
```

```
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
```

```
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
 ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
```

```
        var listRolePoliciesPaginator =
_IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
```

```
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
 policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
```

```
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
```

```
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
```

```
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");
```

```
        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);
```

```
        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
 {bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;
```

```
using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
```

```
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
```

```
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
```

```
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
```

```
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for .NET APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
```

```
 {
   if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

     source ./iam_operations.sh
   fi
 }

 echo_repeat "*" 88
 echo "Welcome to the IAM create user and assume role demo."
 echo
 echo "This demo will create an IAM user, create an IAM role, and apply the role
to the user."
 echo_repeat "*" 88
 echo

 echo -n "Enter a name for a new IAM user: "
 get_input
 user_name=$get_input_result

 local user_arn
 user_arn=$(iam_create_user -u "$user_name")

 # shellcheck disable=SC2181
 if [[ ${?} == 0 ]]; then
   echo "Created demo IAM user named $user_name"
 else
   errecho "$user_arn"
   errecho "The user failed to create. This demo will exit."
   return 1
 fi

 local access_key_response
 access_key_response=$(iam_create_user_access_key -u "$user_name")
 # shellcheck disable=SC2181
 if [[ ${?} != 0 ]]; then
   errecho "The access key failed to create. This demo will exit."
   clean_up "$user_name"
   return 1
 fi

 IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
 local key_name=${access_key_values[0]}
 local key_secret=${access_key_values[1]}
```

```
  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
```

```
 policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
 # shellcheck disable=SC2181
 if [[ ${?} == 0 ]]; then
   echo "Created  IAM policy named $policy_name"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name"
   return 1
 fi

 if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
   echo "Attached policy $policy_arn to role $iam_role_name"
 else
   errecho "The policy failed to attach."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
   return 1
 fi

 local assume_role_policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"sts:AssumeRole\",
                  \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
```

```
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret

  local buckets
  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. This should not have
happened."
  else
    errecho "Because the role with permissions has not been assumed, listing
buckets failed."
  fi

  echo
  echo_repeat "*" 88
  echo "Now assume the role $iam_role_name and list the buckets."
  echo_repeat "*" 88
  echo

  local credentials

  credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Assumed role $iam_role_name"
  else
    errecho "Failed to assume role."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
```

```
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  IFS=$'\t ' read -r -a credentials <<<"$credentials"

  export AWS_ACCESS_KEY_ID=${credentials[0]}
  export AWS_SECRET_ACCESS_KEY=${credentials[1]}
  # bashsupport disable=BP2001
  export AWS_SESSION_TOKEN=${credentials[2]}

  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
"$assume_role_policy_arn"
```

```
  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

IAMFungsi yang digunakan dalam skenario ini.

```
###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
```

```
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
```

```
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
```

```
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
```

```
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-policy operation failed.\n$response"
```

```
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
```

```
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
```

```
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
```

```
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
```

```
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
  return 1
fi

iecho "delete-policy response:$response"
iecho

return 0
```

```
}

########################################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
```

```
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

########################################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.
```

```
 # bashsupport disable=BP5008
 function usage() {
   echo "function iam_delete_access_key"
   echo "Deletes an WS Identity and Access Management (IAM) access key for the
specified IAM user"
   echo "  -u user_name    The name of the user."
   echo "  -k access_key   The access key to delete."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "u:k:h" option; do
   case "${option}" in
     u) user_name="${OPTARG}" ;;
     k) access_key="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$user_name" ]]; then
   errecho "ERROR: You must provide a username with the -u parameter."
   usage
   return 1
 fi

 if [[ -z "$access_key" ]]; then
   errecho "ERROR: You must provide an access key with the -k parameter."
   usage
   return 1
 fi

 iecho "Parameters:\n"
 iecho "    Username:   $user_name"
 iecho "    Access key:   $access_key"
 iecho ""
```

```
  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi
```

```
  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Untuk API detailnya, lihat topik berikut di Referensi AWS CLI Perintah.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
```

```
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);
```

```
        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
std::endl;
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);
```

```
        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
```

```
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
```

```
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }
```

```
   // 5. List objects in the bucket (This should fail).
   {
       Aws::S3::S3Client s3Client(
               Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                         credentials.GetSecretAccessKey(),
                                         credentials.GetSessionToken()),
               Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
               clientConfig);
       Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
       if (!listBucketsOutcome.IsSuccess()) {
           if (listBucketsOutcome.GetError().GetErrorType() !=
               Aws::S3::S3Errors::ACCESS_DENIED) {
               std::cerr << "Could not lists buckets. " <<
                         listBucketsOutcome.GetError().GetMessage() <<
std::endl;
           }
           else {
               std::cout
                       << "Access to list buckets denied because privileges have
not been applied."
                       << std::endl;
           }
       }
       else {
           std::cerr
                   << "Successfully retrieved bucket lists when this should not
happen."
                   << std::endl;
       }
   }

   // 6. Attach the policy to the role.
   {
       Aws::IAM::Model::AttachRolePolicyRequest request;
       request.SetRoleName(role.GetRoleName());
       request.WithPolicyArn(policy.GetArn());

       Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
               request);
       if (!outcome.IsSuccess()) {
           std::cerr << "Error creating policy. " <<
                     outcome.GetError().GetMessage() << std::endl;
```

```
            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
```

```
                        << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());
```

```
            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
```

```
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for C++ APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Jalankan skenario interaktif di penggugah/prompt perintah.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
```

```
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
```

```
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
```

```
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
```

```
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
```

```
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
```

```
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
```

```
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
 *key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
  log.Println(strings.Repeat("-", 88))
 }

}
```

Tentukan struct yang membungkus tindakan akun.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
```

```
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Tentukan struct yang membungkus tindakan kebijakan.

```
// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
```

```
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
```

```
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

```
// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan peran.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

```
// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
```

```
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
```

```
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
```

```
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan pengguna.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
```

```
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
```

```
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
```

```
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
```

```
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Go APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM pengguna.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
```

```
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
 assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
 objects are read.\s
                """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
```

```
        String policyName = args[1];
        String roleName = args[2];
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
```

```
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();
```

```
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
   public static void deleteRole(IamClient iam, String roleName, String polArn)
{

       try {
           // First the policy needs to be detached.
           DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                   .policyArn(polArn)
                   .roleName(roleName)
                   .build();

           iam.detachRolePolicy(rolePolicyRequest);

           // Delete the policy.
           DeletePolicyRequest request = DeletePolicyRequest.builder()
                   .policyArn(polArn)
                   .build();

           iam.deletePolicy(request);
           System.out.println("*** Successfully deleted " + polArn);

           // Delete the role.
           DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           iam.deleteRole(roleRequest);
           System.out.println("*** Successfully deleted " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void deleteKey(IamClient iam, String username, String
accessKey) {
       try {
           DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                   .accessKeyId(accessKey)
                   .userName(username)
                   .build();

           iam.deleteAccessKey(request);
```

```
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Java 2.x APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";
```

```
/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
```

```
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
(AWS STS).
  // It's not best practice to use access keys. For more information, see
https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
```

```
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
      ),
  );

  if (!Role) {
    throw new Error("Role not created");
  }

  // Create a policy that allows the user to list S3 buckets.
  const { Policy: listBucketPolicy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Action: ["s3:ListAllMyBuckets"],
            Resource: "*",
          },
        ],
      }),
```

```
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
```

```
    credentials: {
      accessKeyId: Credentials.AccessKeyId,
      secretAccessKey: Credentials.SecretAccessKey,
      sessionToken: Credentials.SessionToken,
    },
  });

  // List the S3 buckets again.
  // Retry the list buckets operation until it succeeds. AccessDenied might
  // be thrown while the role policy is still stabilizing.
  await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
    listBuckets(s3Client),
  );

  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
```

```
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for JavaScript APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM pengguna.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
```

```
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue: String =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
```

```
            "}"

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
```

```
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient =
        StsClient {
```

```
            region = "us-east-1"
        }

    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials =
        StaticCredentialsProvider {
            accessKeyId = key
            secretAccessKey = secKey
            sessionToken = secToken
        }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 =
        S3Client {
            credentialsProvider = staticCredentials
            region = "us-east-1"
        }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}
```

```
suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}
```

```
@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk API referensi Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';
```

```
use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("--------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("--------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
```

```
$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}
```

```
$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for PHP APIReferensi.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
```

```
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
```

```
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
```

```
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
```

```
    Assumes a role that grants permission to list the Amazon S3 buckets in the
account.
    Uses the temporary credentials from the role to list the buckets that are
owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
```

```
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )
```

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
```

```
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
```

```
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
```

```
 # @param policy_name [String] The name to give the policy.
 # @param user [Aws::IAM::User] The user that owns the policy.
 # @param role [Aws::IAM::Role] The role that can be assumed.
 # @return [Aws::IAM::UserPolicy] The newly created policy.
 def create_user_policy(policy_name, user, role)
   policy_document = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Action: 'sts:AssumeRole',
       Resource: role.arn
     }]
   }.to_json
   @iam_client.put_user_policy(
     user_name: user.user_name,
     policy_name: policy_name,
     policy_document: policy_document
   )
   puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
```

```
    count = 10
    s3_resource.buckets.each do |bucket|
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == 'AccessDenied'
      puts('Attempt to list buckets with no permissions: AccessDenied.')
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
  # This is separated into a factory function so that it can be mocked for unit
testing.
  #
  # @param key_id [String] The ID of the access key used by the STS client.
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
```

```
  end

  # Deletes a role. If the role has policies attached, they are detached and
  # deleted before the role is deleted.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    @iam_client.list_attached_role_policies(role_name:
 role_name).attached_policies.each do |policy|
      @iam_client.detach_role_policy(role_name: role_name, policy_arn:
 policy.policy_arn)
      @iam_client.delete_policy(policy_arn: policy.policy_arn)
      @logger.info("Detached and deleted policy #{policy.policy_name}.")
    end
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Role deleted: #{role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end
```

```
# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Ruby APIReferensi.

- AttachRolePolicy
- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
```

```
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
```

```
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
```

```
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
```

```
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
```

```
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk API referensi Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Tindakan untuk IAM menggunakan AWS SDKs

Contoh kode berikut menunjukkan bagaimana melakukan IAM tindakan individu dengan AWS SDKs. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Kutipan ini menyebut IAM API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Anda dapat melihat tindakan dalam konteks diSkenario untuk IAM menggunakan AWS SDKs .

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkap, lihat AWS Identity and Access Management APIReferensi.

Contoh

- Gunakan AddClientIdToOpenIdConnectProvider dengan CLI
- Gunakan AddRoleToInstanceProfile dengan CLI
- Gunakan AddUserToGroup dengan AWS SDK atau CLI
- Gunakan AttachGroupPolicy dengan CLI
- Gunakan AttachRolePolicy dengan AWS SDK atau CLI
- Gunakan AttachUserPolicy dengan AWS SDK atau CLI
- Gunakan ChangePassword dengan CLI
- Gunakan CreateAccessKey dengan AWS SDK atau CLI
- Gunakan CreateAccountAlias dengan AWS SDK atau CLI
- Gunakan CreateGroup dengan AWS SDK atau CLI
- Gunakan CreateInstanceProfile dengan AWS SDK atau CLI

- Gunakan CreateLoginProfile dengan CLI
- Gunakan CreateOpenIdConnectProvider dengan a CLI
- Gunakan CreatePolicy dengan AWS SDK atau CLI
- Gunakan CreatePolicyVersion dengan AWS SDK atau CLI
- Gunakan CreateRole dengan AWS SDK atau CLI
- Gunakan CreateSAMLProvider dengan AWS SDK atau CLI
- Gunakan CreateServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan CreateUser dengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevice dengan CLI
- Gunakan DeactivateMfaDevice dengan CLI
- Gunakan DeleteAccessKey dengan AWS SDK atau CLI
- Gunakan DeleteAccountAlias dengan AWS SDK atau CLI
- Gunakan DeleteAccountPasswordPolicy dengan CLI
- Gunakan DeleteGroup dengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicy dengan AWS SDK atau CLI
- Gunakan DeleteInstanceProfile dengan AWS SDK atau CLI
- Gunakan DeleteLoginProfile dengan CLI
- Gunakan DeleteOpenIdConnectProvider dengan CLI
- Gunakan DeletePolicy dengan AWS SDK atau CLI
- Gunakan DeletePolicyVersion dengan a CLI
- Gunakan DeleteRole dengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundary dengan CLI
- Gunakan DeleteRolePolicy dengan AWS SDK atau CLI
- Gunakan DeleteSAMLProvider dengan AWS SDK atau CLI
- Gunakan DeleteServerCertificate dengan AWS SDK atau CLI
- Gunakan DeleteServiceLinkedRole dengan AWS SDK atau CLI
- Gunakan DeleteSigningCertificate dengan CLI
- Gunakan DeleteUser dengan AWS SDK atau CLI

- Gunakan DeleteUserPermissionsBoundary dengan CLI
- Gunakan DeleteUserPolicy dengan AWS SDK atau CLI
- Gunakan DeleteVirtualMfaDevice dengan CLI
- Gunakan DetachGroupPolicy dengan CLI
- Gunakan DetachRolePolicy dengan AWS SDK atau CLI
- Gunakan DetachUserPolicy dengan AWS SDK atau CLI
- Gunakan EnableMfaDevice dengan CLI
- Gunakan GenerateCredentialReport dengan AWS SDK atau CLI
- Gunakan GenerateServiceLastAccessedDetails dengan CLI
- Gunakan GetAccessKeyLastUsed dengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetails dengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicy dengan AWS SDK atau CLI
- Gunakan GetAccountSummary dengan AWS SDK atau CLI
- Gunakan GetContextKeysForCustomPolicy dengan CLI
- Gunakan GetContextKeysForPrincipalPolicy dengan CLI
- Gunakan GetCredentialReport dengan AWS SDK atau CLI
- Gunakan GetGroup dengan CLI
- Gunakan GetGroupPolicy dengan CLI
- Gunakan GetInstanceProfile dengan CLI
- Gunakan GetLoginProfile dengan a CLI
- Gunakan GetOpenIdConnectProvider dengan CLI
- Gunakan GetPolicy dengan AWS SDK atau CLI
- Gunakan GetPolicyVersion dengan AWS SDK atau CLI
- Gunakan GetRole dengan AWS SDK atau CLI
- Gunakan GetRolePolicy dengan CLI
- Gunakan GetSamlProvider dengan CLI
- Gunakan GetServerCertificate dengan AWS SDK atau CLI
- Gunakan GetServiceLastAccessedDetails dengan CLI

- Gunakan GetServiceLastAccessedDetailsWithEntities dengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatus dengan AWS SDK atau CLI
- Gunakan GetUser dengan AWS SDK atau CLI
- Gunakan GetUserPolicy dengan a CLI
- Gunakan ListAccessKeys dengan AWS SDK atau CLI
- Gunakan ListAccountAliases dengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPolicies dengan a CLI
- Gunakan ListAttachedRolePolicies dengan AWS SDK atau CLI
- Gunakan ListAttachedUserPolicies dengan CLI
- Gunakan ListEntitiesForPolicy dengan a CLI
- Gunakan ListGroupPolicies dengan CLI
- Gunakan ListGroups dengan AWS SDK atau CLI
- Gunakan ListGroupsForUser dengan CLI
- Gunakan ListInstanceProfiles dengan CLI
- Gunakan ListInstanceProfilesForRole dengan CLI
- Gunakan ListMfaDevices dengan CLI
- Gunakan ListOpenIdConnectProviders dengan CLI
- Gunakan ListPolicies dengan AWS SDK atau CLI
- Gunakan ListPolicyVersions dengan CLI
- Gunakan ListRolePolicies dengan AWS SDK atau CLI
- Gunakan ListRoleTags dengan CLI
- Gunakan ListRoles dengan AWS SDK atau CLI
- Gunakan ListSAMLProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificates dengan AWS SDK atau CLI
- Gunakan ListSigningCertificates dengan CLI
- Gunakan ListUserPolicies dengan AWS SDK atau CLI
- Gunakan ListUserTags dengan CLI
- Gunakan ListUsers dengan AWS SDK atau CLI

- Gunakan ListVirtualMfaDevices dengan a CLI
- Gunakan PutGroupPolicy dengan AWS SDK atau CLI
- Gunakan PutRolePermissionsBoundary dengan CLI
- Gunakan PutRolePolicy dengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundary dengan CLI
- Gunakan PutUserPolicy dengan AWS SDK atau CLI
- Gunakan RemoveClientIdFromOpenIdConnectProvider dengan CLI
- Gunakan RemoveRoleFromInstanceProfile dengan CLI
- Gunakan RemoveUserFromGroup dengan AWS SDK atau CLI
- Gunakan ResyncMfaDevice dengan CLI
- Gunakan SetDefaultPolicyVersion dengan CLI
- Gunakan TagRole dengan CLI
- Gunakan TagUser dengan CLI
- Gunakan UntagRole dengan CLI
- Gunakan UntagUser dengan CLI
- Gunakan UpdateAccessKey dengan AWS SDK atau CLI
- Gunakan UpdateAccountPasswordPolicy dengan CLI
- Gunakan UpdateAssumeRolePolicy dengan CLI
- Gunakan UpdateGroup dengan CLI
- Gunakan UpdateLoginProfile dengan CLI
- Gunakan UpdateOpenIdConnectProviderThumbprint dengan CLI
- Gunakan UpdateRole dengan CLI
- Gunakan UpdateRoleDescription dengan CLI
- Gunakan UpdateSamlProvider dengan CLI
- Gunakan UpdateServerCertificate dengan AWS SDK atau CLI
- Gunakan UpdateSigningCertificate dengan CLI
- Gunakan UpdateUser dengan AWS SDK atau CLI
- Gunakan UploadServerCertificate dengan AWS SDK atau CLI
- Gunakan UploadSigningCertificate dengan CLI

Gunakan **`AddClientIdToOpenIdConnectProvider`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AddClientIdToOpenIdConnectProvider`.

CLI

AWS CLI

Untuk menambahkan ID klien (audiens) ke penyedia Open-ID Connect (OIDC)

`add-client-id-to-open-id-connect-provider`Perintah berikut menambahkan ID klien `my-application-ID` ke OIDC penyedia bernama`server.example.com`.

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Perintah ini tidak menghasilkan output.

Untuk membuat OIDC penyedia, gunakan `create-open-id-connect-provider` perintah.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat AddClientIdToOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menambahkan ID klien (atau audiens) **`my-application-ID`** ke OIDC penyedia yang ada bernama**`server.example.com`**.

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Untuk API detailnya, lihat AddClientIdToOpenIdConnectProviderdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **AddRoleToInstanceProfile** dengan CLI

Contoh kode berikut menunjukkan cara menggunakanAddRoleToInstanceProfile.

CLI

AWS CLI

Untuk menambahkan peran ke profil instance

add-role-to-instance-profilePerintah berikut menambahkan peran bernama S3Access ke profil instance bernamaWebserver.

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

Perintah ini tidak menghasilkan output.

Untuk membuat profil instance, gunakan create-instance-profile perintah.

Untuk informasi selengkapnya, lihat Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat AddRoleToInstanceProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menambahkan peran bernama **S3Access** ke profil instance yang ada bernama**webserver**. Untuk membuat profil instance, gunakan **New-IAMInstanceProfile** perintah. Setelah Anda membuat profil instance dan mengaitkannya dengan peran menggunakan perintah ini, Anda dapat melampirkannya ke sebuah EC2 instance. Untuk melakukan itu, gunakan **New-EC2Instance** cmdlet dengan **InstanceProfile-Name** parameter **InstanceProfile_Arn** atau untuk meluncurkan instance baru.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Untuk API detailnya, lihat AddRoleToInstanceProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`AddUserToGroup`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AddUserToGroup`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat grup dan tambahkan pengguna

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
```

```
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat AddUserToGroupdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menambahkan pengguna ke IAM grup

`add-user-to-group`Perintah berikut menambahkan nama IAM pengguna Bob ke IAM grup bernama`Admins`.

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menambahkan dan menghapus IAM pengguna di grup AWS IAMpengguna di Panduan Pengguna.

- Untuk API detailnya, lihat AddUserToGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menambahkan nama pengguna **`Bob`** ke grup bernama**`Admins`**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Untuk API detailnya, lihat AddUserToGroupdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`AttachGroupPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AttachGroupPolicy`.

CLI

AWS CLI

Untuk melampirkan kebijakan terkelola ke IAM grup

`attach-group-policy`Perintah berikut melampirkan kebijakan AWS terkelola bernama `ReadOnlyAccess` ke IAM grup bernama`Finance`.

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --group-name Finance
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan terkelola dan kebijakan sebaris di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat AttachGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini melampirkan kebijakan terkelola pelanggan yang diberi nama **`TesterPolicy`** ke IAM grup**`Testers`**. Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi default kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini melampirkan kebijakan AWS terkelola yang dinamai **`AdministratorAccess`** ke IAM grup**`Admins`**. Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Untuk API detailnya, lihat AttachGroupPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`AttachRolePolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AttachRolePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat grup dan tambahkan pengguna
- Kelola peran

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
```

```
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
```

```
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
```

```
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat AttachRolePolicydi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
```

```
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
            return false;
        }

        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
                                return policy.GetPolicyArn() == policyArn;
                        })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
<<
                  roleName << std::endl;
    }

    return outcome.IsSuccess();
```

```
}
```

- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk melampirkan kebijakan terkelola ke IAM peran

`attach-role-policy`Perintah berikut melampirkan kebijakan AWS terkelola bernama `ReadOnlyAccess` ke IAM peran bernama`ReadOnlyRole`.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan terkelola dan kebijakan sebaris di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat AttachRolePolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
```

```
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
import java.util.List;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
```

```
            ListAttachedRolePoliciesRequest request =
 ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
 iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

            // Ensure that the policy is not attached to this role
            String polArn = "";
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
 this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);

            System.out.println("Successfully attached policy " + policyArn +
                    " to role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Lampirkan kebijakan.

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
      }
    });
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun attachIAMRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}
```

```
fun checkList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- Untuk API detailnya, lihat AttachRolePolicy AWSSDKAPIreferensi Kotlin.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
```

```
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola yang diberi nama **`SecurityAudit`** ke IAM peran**`CoSecurityAuditors`**. Pengguna yang menganggap peran tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Untuk API detailnya, lihat AttachRolePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Lampirkan kebijakan ke peran menggunakan objek Kebijakan Boto3.

```
def attach_to_role(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

Lampirkan kebijakan ke peran menggunakan objek Peran Boto3.

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
```

```
            iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
            logger.info("Attached policy %s to role %s.", policy_arn, role_name)
        except ClientError:
            logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
    role_name)
            raise
```

- Untuk API detailnya, lihat AttachRolePolicy AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
```

```
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end
```

```
  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat AttachRolePolicydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
}
```

- Untuk API detailnya, lihat AttachRolePolicy AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func attachRolePolicy(role: String, policyArn: String) async throws {
        let input = AttachRolePolicyInput(
            policyArn: policyArn,
            roleName: role
        )
        do {
            _ = try await client.attachRolePolicy(input: input)
        } catch {
            print("ERROR: Attaching a role policy:", dump(error))
            throw error
        }
```

```
    }
```

- Untuk API detailnya, lihat AttachRolePolicy AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`AttachUserPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AttachUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

CLI

AWS CLI

Untuk melampirkan kebijakan terkelola ke IAM pengguna

`attach-user-policy`Perintah berikut melampirkan kebijakan AWS terkelola bernama `AdministratorAccess` ke IAM pengguna bernama`Alice`.

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan terkelola dan kebijakan sebaris di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat AttachUserPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola bernama **`AmazonCognitoPowerUser`** ke IAM pengguna**`Bob`**. Pengguna langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Untuk API detailnya, lihat AttachUserPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise
```

- Untuk API detailnya, lihat AttachUserPolicy AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
# Attaches a policy to a user
#
# @param user_name [String] The name of the user
# @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_user(user_name, policy_arn)
  @iam_client.attach_user_policy(
    user_name: user_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to user: #{e.message}")
  false
end
```

- Untuk API detailnya, lihat AttachUserPolicydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Untuk API detailnya, lihat AttachUserPolicy AWSSDKAPIreferensi Rust.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ChangePassword`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ChangePassword`.

CLI

AWS CLI

Untuk mengubah kata sandi untuk IAM pengguna Anda

Untuk mengubah kata sandi untuk IAM pengguna Anda, kami sarankan menggunakan `--cli-input-json` parameter untuk meneruskan JSON file yang berisi kata sandi lama dan baru Anda. Dengan menggunakan metode ini, Anda dapat menggunakan kata sandi yang kuat dengan karakter non-alfanumerik. Mungkin sulit untuk menggunakan kata sandi dengan karakter non-alfanumerik ketika Anda meneruskannya sebagai parameter baris perintah. Untuk menggunakan `--cli-input-json` parameter, mulailah dengan menggunakan `change-password` perintah dengan `--generate-cli-skeleton` parameter, seperti pada contoh berikut.

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

Perintah sebelumnya membuat JSON file bernama change-password.json yang dapat Anda gunakan untuk mengisi kata sandi lama dan baru Anda. Misalnya, file mungkin terlihat seperti berikut ini.

```
{
    "OldPassword": "3s0K_;xh4~8XXI",
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

Selanjutnya, untuk mengubah kata sandi Anda, gunakan `change-password` perintah lagi, kali ini melewati `--cli-input-json` parameter untuk menentukan JSON file Anda. `change-password`Perintah berikut menggunakan `--cli-input-json` parameter dengan JSON file bernama change-password.json.

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

Perintah ini tidak menghasilkan output.

Perintah ini hanya dapat dipanggil oleh IAM pengguna. Jika perintah ini disebut menggunakan kredensi AWS akun (root), perintah mengembalikan kesalahan`InvalidUserType`.

Untuk informasi selengkapnya, lihat Cara IAM pengguna mengubah kata sandi mereka sendiri di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ChangePassworddi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengubah kata sandi untuk pengguna yang menjalankan perintah. Perintah ini hanya dapat dipanggil oleh IAM pengguna. Jika perintah ini dipanggil ketika Anda masuk dengan kredensi AWS akun (root), perintah mengembalikan kesalahan. **`InvalidUserType`**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Untuk API detailnya, lihat ChangePassworddi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateAccessKey`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateAccessKey`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat grup dan tambahkan pengguna
- Buat pengguna read-only dan read-write
- Kelola kunci akses

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
```

```
            UserName = userName,
        });

        return response.AccessKey;

    }
```

- Untuk API detailnya, lihat CreateAccessKeydi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- Untuk API detailnya, lihat CreateAccessKeydi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
```

```
    request.SetUserName(userName);

    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- Untuk API detailnya, lihat CreateAccessKeydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk membuat kunci akses bagi IAM pengguna

`create-access-key`Perintah berikut membuat kunci akses (ID kunci akses dan kunci akses rahasia) untuk IAM pengguna bernama`Bob`.

```
aws iam create-access-key \
    --user-name Bob
```

Output:

```
{
    "AccessKey": {
```

```
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

Simpan kunci akses rahasia di lokasi yang aman. Jika hilang, itu tidak dapat dipulihkan, dan Anda harus membuat kunci akses baru.

Untuk informasi selengkapnya, lihat Mengelola kunci akses untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat CreateAccessKeydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
```

```
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- Untuk API detailnya, lihat CreateAccessKeydi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccessKey {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                   <user>\s

                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }

    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk API detailnya, lihat CreateAccessKeydi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat kunci akses.

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat CreateAccessKeydi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
```

```
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat CreateAccessKeydi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMAccessKey(user: String?): String {
    val request =
        CreateAccessKeyRequest {
            userName = user
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Untuk API detailnya, lihat CreateAccessKey AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat kunci akses baru dan secret access key pair dan menugaskannya ke pengguna**David**. Pastikan Anda menyimpan **AccessKeyId** dan **SecretAccessKey** nilai ke file karena ini adalah satu-satunya waktu Anda dapat memperoleh file**SecretAccessKey**. Anda tidak dapat mengambilnya di lain waktu. Jika Anda kehilangan kunci rahasia, Anda harus membuat access key pair baru.

```
New-IAMAccessKey -UserName David
```

Output:

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Untuk API detailnya, lihat CreateAccessKeydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
```

```
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- Untuk API detailnya, lihat CreateAccessKey AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
```

```
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
```

```
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat CreateAccessKeydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;
```

```
    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- Untuk API detailnya, lihat CreateAccessKey AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createAccessKey(userName: String) async throws ->
 IAMClientTypes.AccessKey {
        let input = CreateAccessKeyInput(
            userName: userName
        )
        do {
```

```
            let output = try await iamClient.createAccessKey(input: input)
            guard let accessKey = output.accessKey else {
                throw ServiceHandlerError.keyError
            }
            return accessKey
        } catch {
            print("ERROR: createAccessKey:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat CreateAccessKey AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateAccountAlias`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateAccountAlias`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created account alias " << aliasName <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat CreateAccountAliasdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk membuat alias akun

`create-account-alias`Perintah berikut membuat alias `examplecorp` untuk AWS akun Anda.

```
aws iam create-account-alias \
    --account-alias examplecorp
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat ID AWS akun Anda dan aliasnya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat CreateAccountAliasdi Referensi AWS CLI Perintah.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccountAlias {
    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
```

```
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat CreateAccountAliasdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat alias akun.

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat CreateAccountAliasdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
```

```
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat CreateAccountAliasdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMAccountAlias(alias: String) {
    val request =
        CreateAccountAliasRequest {
            accountAlias = alias
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- Untuk API detailnya, lihat CreateAccountAlias AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengubah alias akun untuk AWS **mycompanyaws** akun Anda. Alamat halaman logon pengguna diarahkan ke panyaws.signin.aws.amazon.com/console. https://mycom Asli URL menggunakan nomor ID akun Anda alih-alih alias (https://

<accountidnumber>.signin.aws.amazon.com/console) terus berfungsi. Namun, semua berbasis alias yang didefinisikan sebelumnya URLs berhenti bekerja.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk API detailnya, lihat CreateAccountAliasdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

- Untuk API detailnya, lihat CreateAccountAlias AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat CreateAccountAliasdi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateGroup`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateGroup`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat grup dan tambahkan pengguna

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }
```

- Untuk API detailnya, lihat CreateGroupdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk membuat IAM grup

`create-group`Perintah berikut membuat IAM grup bernama`Admins`.

```
aws iam create-group \
    --group-name Admins
```

Output:

```
{
    "Group": {
```

```
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

Untuk informasi selengkapnya, lihat Membuat grup IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat CreateGroupdi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk API detailnya, lihat CreateGroupdi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat IAM grup baru bernama**`Developers`**.

```
New-IAMGroup -GroupName Developers
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Untuk API detailnya, lihat CreateGroupdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateInstanceProfile`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateInstanceProfile`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Membangun dan mengelola layanan yang tangguh

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {

       var assumeRoleDoc = "{" +
                                "\"Version\": \"2012-10-17\"," +
                                "\"Statement\": [{" +
                                     "\"Effect\": \"Allow\"," +
                                     "\"Principal\": {" +
                                     "\"Service\": [" +
                                         "\"ec2.amazonaws.com\"" +
                                     "]" +
                                     "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                           "}";

       var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

       var policyArn = "";

       try
       {
           var createPolicyResult = await _amazonIam.CreatePolicyAsync(
```

```
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
```

```
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
```

```
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- Untuk API detailnya, lihat CreateInstanceProfiledi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk membuat profil instance

`create-instance-profile`Perintah berikut menciptakan sebuah contoh profil bernama`Webserver`.

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

Output:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

Untuk menambahkan peran ke profil instance, gunakan `add-role-to-instance-profile` perintah.

Untuk informasi selengkapnya, lihat Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat CreateInstanceProfiledi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
const { InstanceProfile } = await iamClient.send(
  new CreateInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
  }),
);
await waitUntilInstanceProfileExists(
  { client: iamClient },
  { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
);
```

- Untuk API detailnya, lihat CreateInstanceProfiledi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat profil IAM instance baru bernama**`ProfileForDevEC2Instance`**. Anda harus menjalankan **`Add-IAMRoleToInstanceProfile`** perintah secara terpisah untuk mengaitkan profil instance dengan IAM peran yang ada yang memberikan izin ke instance. Terakhir, lampirkan profil instance ke EC2 instance saat Anda meluncurkannya. Untuk melakukan itu, gunakan **`New-EC2Instance`** cmdlet dengan parameter **`InstanceProfile_Arn`** or**`InstanceProfile_Name`**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Untuk API detailnya, lihat CreateInstanceProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Contoh ini membuat profil kebijakan, peran, dan instance dan menautkan semuanya.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.
```

```
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
```

```
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
```

```
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn
```

- Untuk API detailnya, lihat CreateInstanceProfile AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateLoginProfile`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateLoginProfile`.

CLI

AWS CLI

Untuk membuat kata sandi untuk IAM pengguna

Untuk membuat kata sandi bagi IAM pengguna, kami sarankan menggunakan `--cli-input-json` parameter untuk meneruskan JSON file yang berisi kata sandi. Dengan menggunakan metode ini, Anda dapat membuat kata sandi yang kuat dengan karakter non-alfanumerik. Mungkin sulit untuk membuat kata sandi dengan karakter non-alfanumerik ketika Anda meneruskannya sebagai parameter baris perintah.

Untuk menggunakan `--cli-input-json` parameter, mulailah dengan menggunakan `create-login-profile` perintah dengan `--generate-cli-skeleton` parameter, seperti pada contoh berikut.

```
aws iam create-login-profile \
    --generate-cli-skeleton > create-login-profile.json
```

Perintah sebelumnya membuat JSON file bernama create-login-profile .json yang dapat Anda gunakan untuk mengisi informasi untuk perintah berikutnya`create-login-profile`. Sebagai contoh:

```
{
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

Selanjutnya, untuk membuat kata sandi untuk IAM pengguna, gunakan `create-login-profile` perintah lagi, kali ini melewati `--cli-input-json` parameter untuk menentukan JSON file Anda. `create-login-profile`Perintah berikut menggunakan `--cli-input-json` parameter dengan JSON file bernama create-login-profile .json.

```
aws iam create-login-profile \
    --cli-input-json file://create-login-profile.json
```

Output:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

Jika kata sandi baru melanggar kebijakan kata sandi akun, perintah mengembalikan `PasswordPolicyViolation` kesalahan.

Untuk mengubah kata sandi untuk pengguna yang sudah memilikinya, gunakan`update-login-profile`. Untuk menetapkan kebijakan kata sandi untuk akun, gunakan `update-account-password-policy` perintah.

Jika kebijakan kata sandi akun memungkinkan mereka, IAM pengguna dapat mengubah kata sandi mereka sendiri menggunakan `change-password` perintah.

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat CreateLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat kata sandi (sementara) untuk IAM pengguna bernama Bob, dan menetapkan tanda yang mengharuskan pengguna untuk mengubah kata sandi saat **Bob** masuk berikutnya.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Output:

```
CreateDate                  PasswordResetRequired             UserName
----------                  ---------------------             --------
4/14/2015 12:26:30 PM       True                              Bob
```

- Untuk API detailnya, lihat CreateLoginProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateOpenIdConnectProvider`** dengan a CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateOpenIdConnectProvider`.

CLI

AWS CLI

Untuk membuat penyedia OpenID Connect () OIDC

Untuk membuat penyedia OpenID Connect (OIDC), sebaiknya gunakan `--cli-input-json` parameter untuk meneruskan JSON file yang berisi parameter yang diperlukan. Ketika Anda membuat OIDC penyedia, Anda harus melewati penyedia, dan URL harus dimulai dengan`https://`. URL Mungkin sulit untuk meneruskan URL sebagai parameter baris perintah, karena karakter titik dua (:) dan garis miring maju (/) memiliki arti khusus di beberapa lingkungan baris perintah. Menggunakan `--cli-input-json` parameter mengatasi batasan ini.

Untuk menggunakan `--cli-input-json` parameter, mulailah dengan menggunakan `create-open-id-connect-provider` perintah dengan `--generate-cli-skeleton` parameter, seperti pada contoh berikut.

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

Perintah sebelumnya membuat JSON file bernama create-open-id-connect -provider.json yang dapat Anda gunakan untuk mengisi informasi untuk perintah berikutnya. `create-open-id-connect-provider` Sebagai contoh:

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

Selanjutnya, untuk membuat penyedia OpenID Connect (OIDC), gunakan `create-open-id-connect-provider` perintah lagi, kali ini meneruskan `--cli-input-json` parameter untuk menentukan file AndaJSON. `create-open-id-connect-provider`Perintah berikut menggunakan `--cli-input-json` parameter dengan JSON file bernama create-open-id-connect -provider.json.

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

Output:

```
{
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

Untuk informasi selengkapnya tentang OIDC penyedia, lihat Membuat penyedia identitas OpenID Connect (OIDC) di AWS IAMPanduan Pengguna.

Untuk informasi selengkapnya tentang mendapatkan cap jempol untuk OIDC penyedia, lihat Memperoleh cap jempol untuk Penyedia Identitas OpenID Connect di Panduan Pengguna.AWS IAM

- Untuk API detailnya, lihat CreateOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat IAM OIDC penyedia yang terkait dengan layanan penyedia yang OIDC kompatibel yang ditemukan di URL **`https://example.oidcprovider.com`** dan ID klien**`my-testapp-1`**. OIDCPenyedia memasok sidik jari. Untuk mengautentikasi sidik jari, ikuti langkah-langkah di http://docs.aws.amazon. com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint .html.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Output:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk API detailnya, lihat CreateOpenIdConnectProviderdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreatePolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreatePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat grup dan tambahkan pengguna
- Buat pengguna read-only dan read-write
- Kelola kebijakan
- Bekerja dengan Pembuat IAM Kebijakan API

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
```

```
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- Untuk API detailnya, lihat CreatePolicydi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi
```

```
  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}
```

- Untuk API detailnya, lihat CreatePolicydi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
                                      const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));

    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
```

```
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }

    return result;
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
                 << "       \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    }"
                 << "   ]"
                 << "}";

    return stringStream.str();
}
```

- Untuk API detailnya, lihat CreatePolicydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Contoh 1: Untuk membuat kebijakan terkelola pelanggan

Perintah berikut membuat kebijakan terkelola pelanggan bernama`my-policy`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy
```

File tersebut `policy` adalah JSON dokumen di folder saat ini yang memberikan akses baca saja ke `shared` folder dalam bucket Amazon S3 bernama. `my-bucket`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket/shared/*"
            ]
        }
    ]
}
```

Output:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
```

```
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

Untuk informasi selengkapnya tentang menggunakan file sebagai masukan untuk parameter string, lihat Menentukan nilai parameter untuk AWS CLI di Panduan AWS CLI Pengguna.

Contoh 2: Untuk membuat kebijakan yang dikelola pelanggan dengan deskripsi

Perintah berikut membuat kebijakan terkelola pelanggan bernama `my-policy` dengan deskripsi yang tidak dapat diubah:

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for my-bucket"
```

File tersebut `policy.json` adalah JSON dokumen di folder saat ini yang memberikan akses ke semua tindakan Put, List, dan Get untuk bucket Amazon S3 bernama. `my-bucket`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Output:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

Untuk informasi selengkapnya tentang Kebijakan Berbasis Identifikasi, lihat Kebijakan berbasis identitas dan kebijakan berbasis sumber daya di Panduan Pengguna.AWS IAM

Contoh 3: Membuat kebijakan terkelola pelanggan dengan tag

Perintah berikut membuat kebijakan terkelola pelanggan bernama `my-policy` dengan tag. Contoh ini menggunakan flag `--tags` parameter dengan tag JSON -formatted berikut:. `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'` Atau, `--tags` bendera dapat digunakan dengan tag dalam format singkatan:. `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

File tersebut `policy.json` adalah JSON dokumen di folder saat ini yang memberikan akses ke semua tindakan Put, List, dan Get untuk bucket Amazon S3 bernama. `my-bucket`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Output:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {

        ]
    }
}
```

Untuk informasi selengkapnya tentang kebijakan Penandaan, lihat Menandai kebijakan yang dikelola pelanggan di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat CreatePolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
```

```
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Untuk API detailnya, lihat CreatePolicydi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreatePolicy {

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
```

```
        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
 + result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);

            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk API detailnya, lihat CreatePolicydi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat kebijakan.

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyName
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat CreatePolicydi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var myManagedPolicy = {
  Version: "2012-10-17",
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
  PolicyName: "myDynamoDBPolicy",
};
```

```
iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat CreatePolicydi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMPolicy(policyNameVal: String?): String {
    val policyDocumentVal =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}"
```

```
    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- Untuk API detailnya, lihat CreatePolicy AWSSDKAPIreferensi Kotlin.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
               \"Version\": \"2012-10-17\",
               \"Statement\": [{
                   \"Effect\": \"Allow\",
                   \"Action\": \"s3:ListAllMyBuckets\",
                   \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    public function createPolicy(string $policyName, string $policyDocument)
    {
```

```
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- Untuk API detailnya, lihat CreatePolicydi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat IAM kebijakan baru di AWS akun saat ini bernama **`MySamplePolicy`** File **`MySamplePolicy.json`** menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses file JSON kebijakan.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Output:

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Untuk API detailnya, lihat CreatePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
```

```
        return policy
```

- Untuk API detailnya, lihat CreatePolicy AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
```

```
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat CreatePolicydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
) -> Result<Policy, iamError> {
```

```
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
    Ok(policy.policy.unwrap())
}
```

- Untuk API detailnya, lihat CreatePolicy AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: createPolicy:", dump(error))
            throw error
        }
```

```
    }
```

- Untuk API detailnya, lihat CreatePolicy AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreatePolicyVersion`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreatePolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan

CLI

AWS CLI

Untuk membuat versi baru dari kebijakan terkelola

Contoh ini membuat `v2` versi baru dari IAM kebijakan yang ARN ada `arn:aws:iam::123456789012:policy/MyPolicy` dan menjadikannya versi default.

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

Output:

```
{
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
```

```
    }
}
```

Untuk informasi selengkapnya, lihat IAMKebijakan pembuatan versi di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat CreatePolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat versi “v2" baru dari IAM kebijakan yang ARN ada **`arn:aws:iam::123456789012:policy/MyPolicy`** dan menjadikannya versi default. **`NewPolicyVersion.json`**File tersebut menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses file JSON kebijakan.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Output:

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
4/15/2015 10:54:54 AM                                        True
          v2
```

- Untuk API detailnya, lihat CreatePolicyVersiondi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
```

```
        return policy_version
```

- Untuk API detailnya, lihat CreatePolicyVersion AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateRole`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateRole`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat grup dan tambahkan pengguna
- Kelola peran

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
/// <summary>
/// Create a new IAM role.
/// </summary>
/// <param name="roleName">The name of the IAM role.</param>
/// <param name="rolePolicyDocument">The name of the IAM policy document
/// for the new role.</param>
/// <returns>The Amazon Resource Name (ARN) of the role.</returns>
```

```
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- Untuk API detailnya, lihat CreateRoledi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
```

```
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat CreateRoledi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::createIamRole(
        const Aws::String &roleName,
```

```
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat CreateRoledi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Contoh 1: Untuk membuat IAM peran

`create-role`Perintah berikut membuat peran bernama `Test-Role` dan melampirkan kebijakan kepercayaan padanya.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Output:

```
{
```

```
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

Kebijakan kepercayaan didefinisikan sebagai JSON dokumen dalam file test-role-trust-policy.json. (Nama file dan ekstensi tidak memiliki signifikansi.) Kebijakan kepercayaan harus menentukan kepala sekolah.

Untuk melampirkan kebijakan izin ke peran, gunakan `put-role-policy` perintah.

Untuk informasi selengkapnya, lihat Membuat IAM peran di Panduan AWS IAM Pengguna.

Contoh 2: Untuk membuat IAM peran dengan durasi sesi maksimum yang ditentukan

`create-role`Perintah berikut membuat peran bernama `Test-Role` dan menetapkan durasi sesi maksimum 7200 detik (2 jam).

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

Output:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
```

```
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

Untuk informasi selengkapnya, lihat Memodifikasi durasi sesi maksimum peran (AWS API) di Panduan AWS IAM Pengguna.

Contoh 3: Untuk membuat IAM Peran dengan tag

Perintah berikut membuat IAM Peran `Test-Role` dengan tag. Contoh ini menggunakan flag `--tags` parameter dengan tag JSON -formatted berikut:. `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'` Atau, `--tags` bendera dapat digunakan dengan tag dalam format singkatan:. `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Output:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
```

```
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Untuk informasi selengkapnya, lihat Memberi tag IAM peran di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat CreateRoledi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Untuk API detailnya, lihat CreateRoledi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s

                Where:
                    rolename - The name of the role to create.\s
```

```
                    fileLocation - The location of the JSON document that
represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
fileLocation) throws Exception {
        try {
            JSONObject jsonObject = (JSONObject)
readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static Object readJsonSimpleDemo(String filename) throws Exception {
```

```
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- Untuk API detailnya, lihat CreateRoledi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat peran.

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
          },
          Action: "sts:AssumeRole",
        },
      ],
```

```
    }),
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Untuk API detailnya, lihat CreateRoledi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
     * @throws AwsException
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
```

```
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- Untuk API detailnya, lihat CreateRoledi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat peran baru bernama **MyNewRole** dan melampirkan padanya kebijakan yang ditemukan dalam file**NewRoleTrustPolicy.json**. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file JSON kebijakan. Dokumen kebijakan yang ditampilkan dalam output URL dikodekan. Itu diterjemahkan dalam contoh ini dengan. **UrlDecode** NETmetode.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Output:

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                           %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                           %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                           %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                           %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
```

```
                              %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                              %20%5D%0D%0A%7D
CreateDate               : 4/15/2015 11:04:23 AM
Path                     : /
RoleId                   : V5PAJI2KPN4EAEXAMPLE1
RoleName                 : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Untuk API detailnya, lihat CreateRoledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
```

```
    :return: The newly created role.
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

- Untuk API detailnya, lihat CreateRole AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
# Creates a role and attaches policies to it.
#
```

```
  # @param role_name [String] The name of the role.
  # @param assume_role_policy_document [Hash] The trust relationship policy
document.
  # @param policy_arns [Array<String>] The ARNs of the policies to attach.
  # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
  def create_role(role_name, assume_role_policy_document, policy_arns)
    response = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: assume_role_policy_document.to_json
    )
    role_arn = response.role.arn

    policy_arns.each do |policy_arn|
      @iam_client.attach_role_policy(
        role_name: role_name,
        policy_arn: policy_arn
      )
    end

    role_arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating role: #{e.message}")
    nil
  end
```

- Untuk API detailnya, lihat CreateRoledi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
```

```
    role_policy_document: &str,
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- Untuk API detailnya, lihat CreateRole AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createRole(name: String, policyDocument: String) async throws ->
 String {
        let input = CreateRoleInput(
            assumeRolePolicyDocument: policyDocument,
            roleName: name
        )
        do {
            let output = try await client.createRole(input: input)
```

```
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            guard let id = role.roleId else {
                throw ServiceHandlerError.noSuchRole
            }
            return id
        } catch {
            print("ERROR: createRole:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat CreateRole AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateSAMLProvider`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateSAMLProvider`.

CLI

AWS CLI

Untuk membuat SAML penyedia

Contoh ini membuat SAML penyedia baru IAM bernama`MySAMLProvider`. Ini dijelaskan oleh dokumen SAML metadata yang ditemukan dalam file. `SAMLMetaData.xml`

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

Output:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia IAM SAML identitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat C reateSAMLProvider di Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import * as path from "node:path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
 */
const sampleMetadataDocument = readFileSync(
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
```

```
    Name: providerName,
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk API detailnya, lihat C reateSAMLProvider di AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat entitas SAML penyedia baru diIAM. Ini dinamai **`MySAMLProvider`** dan dijelaskan oleh dokumen SAML metadata yang ditemukan dalam file**`SAMLMetaData.xml`**, yang diunduh secara terpisah dari situs web penyedia SAML layanan.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Output:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Untuk API detailnya, lihat C reateSAMLProvider di AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateServiceLinkedRole`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateServiceLinkedRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- Untuk API detailnya, lihat CreateServiceLinkedRoledi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk membuat peran terkait layanan

`create-service-linked-role`Contoh berikut membuat peran terkait layanan untuk AWS layanan tertentu dan melampirkan deskripsi yang ditentukan.

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

Output:

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat CreateServiceLinkedRoledi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Untuk API detailnya, lihat CreateServiceLinkedRoledi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat peran terkait layanan.

```
import {
  CreateServiceLinkedRoleCommand,
  GetRoleCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });
  try {
    const response = await client.send(command);
    console.log(response);
    return response;
  } catch (caught) {
    if (
```

```
      caught instanceof Error &&
      caught.name === "InvalidInputException" &&
      caught.message.includes(
        "Service role name AWSServiceRoleForElasticBeanstalk has been taken in
 this account",
      )
    ) {
      console.warn(caught.message);
      return client.send(
        new GetRoleCommand({ RoleName: "AWSServiceRoleForElasticBeanstalk" }),
      );
    }
    throw caught;
  }
};
```

- Untuk API detailnya, lihat CreateServiceLinkedRoledi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
```

```
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- Untuk API detailnya, lihat CreateServiceLinkedRoledi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat peran servicelinked untuk layanan penskalaan otomatis.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Untuk API detailnya, lihat CreateServiceLinkedRoledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
```

```
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
service_name)
        raise
    else:
        return role
```

- Untuk API detailnya, lihat CreateServiceLinkedRole AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates a service-linked role
 #
 # @param service_name [String] The service name to create the role for.
 # @param description [String] The description of the service-linked role.
 # @param suffix [String] Suffix for customizing role name.
 # @return [String] The name of the created role
 def create_service_linked_role(service_name, description, suffix)
   response = @iam_client.create_service_linked_role(
     aws_service_name: service_name, description: description, custom_suffix:
suffix
   )
   role_name = response.role.role_name
   @logger.info("Created service-linked role #{role_name}.")
```

```
    role_name
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't create service-linked role for #{service_name}.
 Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk API detailnya, lihat CreateServiceLinkedRoledi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
        .send()
        .await?;

    Ok(response)
}
```

- Untuk API detailnya, lihat CreateServiceLinkedRole AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createServiceLinkedRole(service: String, suffix: String? = nil,
 description: String?)
                    async throws -> IAMClientTypes.Role {
        let input = CreateServiceLinkedRoleInput(
            awsServiceName: service,
            customSuffix: suffix,
            description: description
        )
        do {
            let output = try await client.createServiceLinkedRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            print("ERROR: createServiceLinkedRole:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat CreateServiceLinkedRole AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateUser`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateUser`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat grup dan tambahkan pengguna
- Buat pengguna read-only dan read-write

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Untuk API detailnya, lihat CreateUserdi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
```

```
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""
```

```
  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat CreateUserdi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateUserRequest create_request;
    create_request.SetUserName(userName);
```

```
    auto create_outcome = iam.CreateUser(create_request);
    if (!create_outcome.IsSuccess()) {
        std::cerr << "Error creating IAM user " << userName << ":" <<
                  create_outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created IAM user " << userName << std::endl;
    }

    return create_outcome.IsSuccess();
```

- Untuk API detailnya, lihat CreateUserdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Contoh 1: Untuk membuat IAM pengguna

`create-user`Perintah berikut membuat IAM pengguna bernama Bob di akun saat ini.

```
aws iam create-user \
    --user-name Bob
```

Output:

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
}
```

Untuk informasi selengkapnya, lihat Membuat IAM pengguna di AWS akun Anda di Panduan AWS IAM Pengguna.

Contoh 2: Untuk membuat IAM pengguna di jalur tertentu

`create-user`Perintah berikut membuat IAM pengguna bernama Bob di jalur yang ditentukan.

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

Output:

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat IAMpengenal di Panduan AWS IAM Pengguna.

Contoh 3: Untuk Membuat IAM Pengguna dengan tag

`create-user`Perintah berikut membuat IAM pengguna bernama Bob dengan tag. Contoh ini menggunakan flag `--tags` parameter dengan tag JSON -formatted berikut:. `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'` Atau, `--tags` bendera dapat digunakan dengan tag dalam format singkatan:. `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Output:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
```

```
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Untuk informasi selengkapnya, lihat Menandai IAM pengguna di Panduan AWS IAM Pengguna.

Contoh 3: Untuk membuat IAM pengguna dengan batas izin yang ditetapkan

`create-user`Perintah berikut membuat IAM pengguna bernama Bob dengan batas izin AmazonS3. FullAccess

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Output:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
    }
}
```

Untuk informasi selengkapnya, lihat Batas izin untuk IAM entitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat CreateUserdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

- Untuk API detailnya, lihat CreateUserdi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
import software.amazon.awssdk.services.iam.model.GetUserResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
 iamWaiter.waitUntilUserExists(userRequest);

 waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk API detailnya, lihat CreateUserdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna.

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat CreateUserdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
    );
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat CreateUserdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Untuk API detailnya, lihat CreateUser AWSSDKAPIreferensi Kotlin.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";


    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
            'UserName' => $name,
```

```
        ]);

        return $result['User'];
    }
```

- Untuk API detailnya, lihat CreateUserdi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat IAM pengguna bernama**Bob**. Jika Bob perlu masuk ke AWS konsol, maka Anda harus menjalankan perintah secara terpisah **`New-IAMLoginProfile`** untuk membuat profil masuk dengan kata sandi. Jika Bob perlu menjalankan AWS PowerShell atau CLI perintah lintas platform atau melakukan AWS API panggilan, maka Anda harus menjalankan **`New-IAMAccessKey`** perintah secara terpisah untuk membuat kunci akses.

```
New-IAMUser -UserName Bob
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Untuk API detailnya, lihat CreateUserdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user
```

- Untuk API detailnya, lihat CreateUser AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates a user and their login profile
 #
 # @param user_name [String] The name of the user
 # @param initial_password [String] The initial password for the user
 # @return [String, nil] The ID of the user if created, or nil if an error
occurred
 def create_user(user_name, initial_password)
   response = @iam_client.create_user(user_name: user_name)
   @iam_client.wait_until(:user_exists, user_name: user_name)
   @iam_client.create_login_profile(
```

```
      user_name: user_name,
      password: initial_password,
      password_reset_required: true
    )
    @logger.info("User '#{user_name}' created successfully.")
    response.user.user_id
  rescue Aws::IAM::Errors::EntityAlreadyExists
    @logger.error("Error creating user '#{user_name}': user already exists.")
    nil
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating user '#{user_name}': #{e.message}")
    nil
  end
```

- Untuk API detailnya, lihat CreateUserdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;

    Ok(response.user.unwrap())
}
```

- Untuk API detailnya, lihat CreateUser AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createUser(name: String) async throws -> String {
        let input = CreateUserInput(
            userName: name
        )
        do {
            let output = try await client.createUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            guard let id = user.userId else {
                throw ServiceHandlerError.noSuchUser
            }
            return id
        } catch {
            print("ERROR: createUser:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat CreateUser AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`CreateVirtualMfaDevice`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateVirtualMfaDevice`.

CLI

AWS CLI

Untuk membuat MFA perangkat virtual

Contoh ini menciptakan MFA perangkat virtual baru yang disebut`BobsMFADevice`. Ini membuat file yang berisi informasi bootstrap yang disebut `QRCode.png` dan menempatkannya di `C:/` direktori. Metode bootstrap yang digunakan dalam contoh ini adalah`QRCodePNG`.

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
    --bootstrap-method QRCodePNG
```

Output:

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multi-faktor (MFA) AWS di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat CreateVirtualMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat MFA perangkat virtual baru. Baris 2 dan 3 mengekstrak **`Base32StringSeed`** nilai yang dibutuhkan program MFA perangkat lunak virtual untuk membuat akun (sebagai alternatif dari kode QR). Setelah Anda mengkonfigurasi program dengan nilai, dapatkan dua kode otentikasi berurutan dari program. Terakhir, gunakan perintah terakhir untuk menautkan MFA perangkat virtual ke IAM pengguna **`Bob`** dan menyinkronkan akun dengan dua kode otentikasi.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Output:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Contoh 2: Contoh ini membuat MFA perangkat virtual baru. Baris 2 dan 3 mengekstrak **QRCodePNG** nilai dan menuliskannya ke file. Gambar ini dapat dipindai oleh program MFA perangkat lunak virtual untuk membuat akun (sebagai alternatif untuk memasukkan StringSeed nilai Base32 secara manual). Setelah Anda membuat akun di MFA program virtual Anda, dapatkan dua kode otentikasi berurutan dan masukkan dalam perintah terakhir untuk menautkan MFA perangkat virtual ke IAM pengguna **Bob** dan menyinkronkan akun.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Output:

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Untuk API detailnya, lihat CreateVirtualMfaDevicedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeactivateMfaDevice`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeactivateMfaDevice`.

CLI

AWS CLI

Untuk menonaktifkan perangkat MFA

Perintah ini menonaktifkan MFA perangkat virtual dengan ARN `arn:aws:iam::210987654321:mfa/BobsMFADevice` yang terkait dengan pengguna. Bob

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multi-faktor (MFA) AWS di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat DeactivateMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menonaktifkan MFA perangkat keras yang terkait dengan pengguna **`Bob`** yang memiliki nomor **`123456789012`** seri.

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Contoh 2: Perintah ini menonaktifkan MFA perangkat virtual yang terkait dengan pengguna **`David`** yang memiliki file. ARN **`arn:aws:iam::210987654321:mfa/David`** Perhatikan bahwa MFA perangkat virtual tidak dihapus dari akun. Perangkat virtual masih ada dan muncul di output **`Get-IAMVirtualMFADevice`** perintah. Sebelum Anda dapat membuat MFA perangkat virtual baru untuk pengguna yang sama, Anda harus menghapus yang lama dengan menggunakan **`Remove-IAMVirtualMFADevice`** perintah.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Untuk API detailnya, lihat DeactivateMfaDevicedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteAccessKey`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteAccessKey`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat grup dan tambahkan pengguna
- Buat pengguna read-only dan read-write
- Kelola kunci akses

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeleteAccessKeydi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
```

```
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- Untuk API detailnya, lihat DeleteAccessKeydi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat DeleteAccessKeydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk menghapus kunci akses bagi IAM pengguna

`delete-access-key`Perintah berikut menghapus kunci akses yang ditentukan (ID kunci akses dan kunci akses rahasia) untuk IAM pengguna bernamaBob.

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Untuk membuat daftar kunci akses yang ditentukan untuk IAM pengguna, gunakan `list-access-keys` perintah.

Untuk informasi selengkapnya, lihat Mengelola kunci akses untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteAccessKeydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- Untuk API detailnya, lihat DeleteAccessKeydi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessKey>\s

                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
```

```
        }

        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat DeleteAccessKeydi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Hapus tombol akses.

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat DeleteAccessKeydi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
```

```
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat DeleteAccessKeydi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteKey(
    userNameVal: String,
    accessKey: String,
) {
    val request =
        DeleteAccessKeyRequest {
            accessKeyId = accessKey
            userName = userNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Untuk API detailnya, lihat DeleteAccessKey AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus AWS access key pair dengan ID kunci **AKIAIOSFODNN7EXAMPLE** dari nama **Bob** pengguna.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Untuk API detailnya, lihat DeleteAccessKeydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- Untuk API detailnya, lihat DeleteAccessKey AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end
```

```
  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
```

```
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat DeleteAccessKeydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
            }
```

```
        }
    }
    Ok(())
}
```

- Untuk API detailnya, lihat DeleteAccessKey AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                                key: IAMClientTypes.AccessKey) async throws
    {
        let userName: String?

        if user != nil {
            userName = user!.userName
        } else {
            userName = nil
        }

        let input = DeleteAccessKeyInput(
            accessKeyId: key.accessKeyId,
            userName: userName
        )
        do {
            _ = try await iamClient.deleteAccessKey(input: input)
        } catch {
            print("ERROR: deleteAccessKey:", dump(error))
            throw error
```

```
        }
    }
```

- Untuk API detailnya, lihat DeleteAccessKey AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteAccountAlias`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteAccountAlias`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
```

```
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat DeleteAccountAliasdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk menghapus alias akun

`delete-account-alias`Perintah berikut menghapus alias `mycompany` untuk akun saat ini.

```
aws iam delete-account-alias \
    --account-alias mycompany
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat ID AWS akun Anda dan aliasnya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteAccountAliasdi Referensi AWS CLI Perintah.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
```

```
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Untuk API detailnya, lihat DeleteAccountAliasdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Hapus alias akun.

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });
```

```
  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat DeleteAccountAliasdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat DeleteAccountAliasdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteIAMAccountAlias(alias: String) {
    val request =
        DeleteAccountAliasRequest {
            accountAlias = alias
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- Untuk API detailnya, lihat DeleteAccountAlias AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus alias akun dari akun Anda Akun AWS. Halaman login pengguna dengan alias di https://mycom panyaws.signin.aws.amazon.com/console tidak lagi berfungsi. Sebagai gantinya, Anda harus menggunakan yang asli URL dengan nomor Akun AWS ID Anda di https://.signin.aws.amazon.com/console. <accountidnumber>

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk API detailnya, lihat DeleteAccountAliasdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- Untuk API detailnya, lihat DeleteAccountAlias AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat DeleteAccountAliasdi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteAccountPasswordPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteAccountPasswordPolicy`.

CLI

AWS CLI

Untuk menghapus kebijakan kata sandi akun saat ini

`delete-account-password-policy`Perintah berikut menghapus kebijakan kata sandi untuk akun saat ini.

```
aws iam delete-account-password-policy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menyetel kebijakan kata sandi akun untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteAccountPasswordPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus kebijakan kata sandi untuk Akun AWS dan mengatur ulang semua nilai ke default aslinya. Jika kebijakan kata sandi saat ini tidak ada, pesan galat berikut akan muncul: Kebijakan akun dengan nama PasswordPolicy tidak dapat ditemukan.

```
Remove-IAMAccountPasswordPolicy
```

- Untuk API detailnya, lihat DeleteAccountPasswordPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteGroup`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteGroup`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat grup dan tambahkan pengguna

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeleteGroupdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menghapus IAM grup

`delete-group`Perintah berikut menghapus IAM grup bernama`MyTestGroup`.

```
aws iam delete-group \
    --group-name MyTestGroup
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menghapus grup IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteGroupdi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
```

```
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk API detailnya, lihat DeleteGroupdi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus IAM grup bernama**MyTestGroup**. Perintah pertama menghapus semua IAM pengguna yang menjadi anggota grup, dan perintah kedua menghapus IAM grup. Kedua perintah bekerja tanpa ada petunjuk untuk konfirmasi.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Untuk API detailnya, lihat DeleteGroupdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteGroupPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteGroupPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat grup dan tambahkan pengguna

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeleteGroupPolicydi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menghapus kebijakan dari IAM grup

`delete-group-policy`Perintah berikut menghapus kebijakan yang dinamai `ExamplePolicy` dari grup bernama`Admins`.

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk melihat kebijakan yang dilampirkan ke grup, gunakan `list-group-policies` perintah.

Untuk informasi selengkapnya, lihat Mengelola IAM kebijakan di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus kebijakan inline bernama **`TesterPolicy`** dari IAM grup**`Testers`**. Pengguna dalam grup tersebut segera kehilangan izin yang ditentukan dalam kebijakan tersebut.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Untuk API detailnya, lihat DeleteGroupPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteInstanceProfile`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteInstanceProfile`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Membangun dan mengelola layanan yang tangguh

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
   /// <summary>
   /// Detaches a role from an instance profile, detaches policies from the
role,
   /// and deletes all the resources.
   /// </summary>
   /// <param name="profileName">The name of the profile to delete.</param>
   /// <param name="roleName">The name of the role to delete.</param>
   /// <returns>Async task.</returns>
   public async Task DeleteInstanceProfile(string profileName, string roleName)
   {
       try
       {
           await _amazonIam.RemoveRoleFromInstanceProfileAsync(
               new RemoveRoleFromInstanceProfileRequest()
               {
                   InstanceProfileName = profileName,
                   RoleName = roleName
               });
           await _amazonIam.DeleteInstanceProfileAsync(
               new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
           var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
               new ListAttachedRolePoliciesRequest() { RoleName = roleName });
           foreach (var policy in attachedPolicies.AttachedPolicies)
           {
               await _amazonIam.DetachRolePolicyAsync(
                   new DetachRolePolicyRequest()
                   {
                       RoleName = roleName,
```

```
                    PolicyArn = policy.PolicyArn
                });
            // Delete the custom policies only.
            if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
            {
                await _amazonIam.DeletePolicyAsync(
                    new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                    {
                        PolicyArn = policy.PolicyArn
                    });
            }
        }

        await _amazonIam.DeleteRoleAsync(
            new DeleteRoleRequest() { RoleName = roleName });
    }
    catch (NoSuchEntityException)
    {
        Console.WriteLine($"Instance profile {profileName} does not exist.");
    }
}
```

- Untuk API detailnya, lihat DeleteInstanceProfiledi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menghapus profil instance

`delete-instance-profile`Perintah berikut menghapus profil instance bernama`ExampleInstanceProfile`.

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menggunakan profil instans di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteInstanceProfiledi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
const client = new IAMClient({});
await client.send(
  new DeleteInstanceProfileCommand({
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
```

- Untuk API detailnya, lihat DeleteInstanceProfiledi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus profil EC2 instance bernama**`MyAppInstanceProfile`**. Perintah pertama melepaskan peran apa pun dari profil instance, dan kemudian perintah kedua menghapus profil instance.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Untuk API detailnya, lihat DeleteInstanceProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Contoh ini menghapus peran dari profil instance, melepaskan semua kebijakan yang dilampirkan pada peran, dan menghapus semua sumber daya.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
```

```
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
```

```
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
```

- Untuk API detailnya, lihat DeleteInstanceProfile AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteLoginProfile`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteLoginProfile`.

CLI

AWS CLI

Untuk menghapus kata sandi untuk IAM pengguna

`delete-login-profile`Perintah berikut menghapus kata sandi untuk IAM pengguna bernamaBob.

```
aws iam delete-login-profile \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus profil login dari IAM pengguna bernama**Bob**. Ini mencegah pengguna masuk ke konsol. AWS Ini tidak mencegah pengguna menjalankan apa pun AWS CLI, PowerShell, atau API panggilan menggunakan kunci AWS akses yang mungkin masih dilampirkan ke akun pengguna.

```
Remove-IAMLoginProfile -UserName Bob
```

- Untuk API detailnya, lihat DeleteLoginProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteOpenIdConnectProvider`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteOpenIdConnectProvider`.

CLI

AWS CLI

Untuk menghapus penyedia identitas IAM OpenID Connect

Contoh ini menghapus IAM OIDC penyedia yang terhubung ke penyedia`example.oidcprovider.com`.

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat DeleteOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus IAM OIDC penyedia yang terhubung ke penyedia**`example.oidcprovider.com`**. Pastikan Anda memperbarui atau menghapus peran apa pun yang mereferensikan penyedia ini dalam **`Principal`** elemen kebijakan kepercayaan peran tersebut.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk API detailnya, lihat DeleteOpenIdConnectProviderdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeletePolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeletePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat pengguna read-only dan read-write
- Kelola kebijakan

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeletePolicydi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function iecho
#
```

```
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
```

```
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}
```

- Untuk API detailnya, lihat DeletePolicydi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat DeletePolicydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk menghapus IAM kebijakan

Contoh ini menghapus kebijakan yang ARN adaarn:aws:iam::123456789012:policy/MySamplePolicy.

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeletePolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- Untuk API detailnya, lihat DeletePolicydi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Untuk API detailnya, lihat DeletePolicydi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Hapus kebijakan.

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- Untuk API detailnya, lihat DeletePolicydi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
    val request =
        DeletePolicyRequest {
            policyArn = policyARNVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Untuk API detailnya, lihat DeletePolicy AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus kebijakan yang ARN ada**arn:aws:iam::123456789012:policy/MySamplePolicy**. Sebelum Anda dapat menghapus kebijakan, Anda harus terlebih dahulu menghapus semua versi kecuali default dengan menjalankan**Remove-IAMPolicyVersion**. Anda juga harus melepaskan kebijakan dari IAM pengguna, grup, atau peran apa pun.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default, melepaskannya dari semua IAM entitas terlampir, dan akhirnya menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai versi default ke dalam koleksi dan kemudian menghapus setiap kebijakan dalam koleksi. Baris ketiga mengambil semua IAM pengguna, grup, dan peran yang dilampirkan kebijakan. Baris empat hingga enam melepaskan kebijakan dari setiap entitas terlampir. Baris terakhir menggunakan perintah ini untuk menghapus kebijakan terkelola serta versi default yang tersisa. Contohnya termasuk parameter **-Force** sakelar pada baris apa pun yang membutuhkannya untuk menekan permintaan konfirmasi.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Untuk API detailnya, lihat DeletePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- Untuk API detailnya, lihat DeletePolicy AWSSDKReferensi Python (Boto3). API

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
```

```
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Untuk API detailnya, lihat DeletePolicy AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
        let input = DeletePolicyInput(
            policyArn: policy.arn
        )
        do {
            _ = try await iamClient.deletePolicy(input: input)
        } catch {
            print("ERROR: deletePolicy:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat DeletePolicy AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeletePolicyVersion`** dengan a CLI

Contoh kode berikut menunjukkan cara menggunakan`DeletePolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Kembalikan versi kebijakan

CLI

AWS CLI

Untuk menghapus versi kebijakan terkelola

Contoh ini menghapus versi yang diidentifikasi sebagai `v2` dari kebijakan yang ARN ada`arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeletePolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus versi yang diidentifikasi sebagai **`v2`** dari kebijakan ARN yang**`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default dan kemudian menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai default ke dalam koleksi dan kemudian menggunakan perintah ini untuk menghapus setiap kebijakan dalam koleksi. Baris terakhir menghapus kebijakan itu sendiri serta versi default yang tersisa. Perhatikan bahwa agar berhasil menghapus kebijakan terkelola, Anda juga harus melepaskan kebijakan dari pengguna, grup, atau peran apa pun dengan menggunakan perintah**Unregister-IAMUserPolicy**,**Unregister-IAMGroupPolicy**, dan **Unregister-IAMRolePolicy** perintah. Lihat contoh untuk **Remove-IAMPolicy** cmdlet.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Untuk API detailnya, lihat DeletePolicyVersiondi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **DeleteRole** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteRole`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Kelola peran

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeleteRoledi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
```

```
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
```

```
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- Untuk API detailnya, lihat DeleteRoledi Referensi AWS CLI Perintah.

CLI

AWS CLI

Untuk menghapus IAM peran

`delete-role`Perintah berikut menghapus peran bernama`Test-Role`.

```
aws iam delete-role \
    --role-name Test-Role
```

Perintah ini tidak menghasilkan output.

Sebelum dapat menghapus peran, Anda harus menghapus peran tersebut dari profil instance (`remove-role-from-instance-profile`), melepaskan kebijakan terkelola (`detach-role-policy`), dan menghapus kebijakan sebaris apa pun yang dilampirkan ke role (`delete-role-policy`).

Untuk informasi selengkapnya, lihat Membuat IAM peran dan Menggunakan profil instans di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteRoledi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- Untuk API detailnya, lihat DeleteRoledi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Hapus peran.

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Untuk API detailnya, lihat DeleteRoledi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus peran yang dinamai **MyNewRole** dari IAM akun saat ini. Sebelum Anda dapat menghapus peran, Anda harus terlebih dahulu menggunakan **Unregister-IAMRolePolicy** perintah untuk melepaskan kebijakan terkelola apa pun. Kebijakan sebaris dihapus dengan peran tersebut.

```
Remove-IAMRole -RoleName MyNewRole
```

Contoh 2: Contoh ini melepaskan kebijakan terkelola apa pun dari peran bernama **MyNewRole** dan kemudian menghapus peran tersebut. Baris pertama mengambil kebijakan terkelola yang melekat pada peran sebagai koleksi dan kemudian melepaskan setiap kebijakan dalam koleksi dari peran tersebut. Baris kedua menghapus peran itu sendiri. Kebijakan inline dihapus bersama dengan peran.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Untuk API detailnya, lihat DeleteRoledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_role(role_name):
    """
```

```
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- Untuk API detailnya, lihat DeleteRole AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    # Detach and delete attached policies
    @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
      response.attached_policies.each do |policy|
        @iam_client.detach_role_policy({
                                         role_name: role_name,
                                         policy_arn: policy.policy_arn
                                       })
        # Check if the policy is a customer managed policy (not AWS managed)
        unless policy.policy_arn.include?('aws:policy/')
          @iam_client.delete_policy({ policy_arn: policy.policy_arn })
          @logger.info("Deleted customer managed policy #{policy.policy_name}.")
```

```
        end
      end
    end

    # Delete the role
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Deleted role #{role_name}.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't detach policies and delete role #{role_name}. Here's
 why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk API detailnya, lihat DeleteRoledi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
        .await
        .is_err()
    {
        sleep(Duration::from_secs(2)).await;
    }
    Ok(())
}
```

- Untuk API detailnya, lihat DeleteRole AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func deleteRole(role: IAMClientTypes.Role) async throws {
        let input = DeleteRoleInput(
            roleName: role.roleName
        )
        do {
            _ = try await iamClient.deleteRole(input: input)
        } catch {
            print("ERROR: deleteRole:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat DeleteRole AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteRolePermissionsBoundary`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteRolePermissionsBoundary`.

CLI

AWS CLI

Untuk menghapus batas izin dari peran IAM

`delete-role-permissions-boundary`Contoh berikut menghapus batas izin untuk peran yang ditentukan. IAM Untuk menerapkan batas izin ke peran, gunakan perintah. `put-role-permissions-boundary`

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteRolePermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan pada peranIAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Untuk API detailnya, lihat DeleteRolePermissionsBoundarydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteRolePolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteRolePolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeleteRolePolicydi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menghapus kebijakan dari IAM peran

`delete-role-policy`Perintah berikut menghapus kebijakan yang dinamai `ExamplePolicy` dari peran bernama`Test-Role`.

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteRolePolicydi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
  });
  return client.send(command);
};
```

- Untuk API detailnya, lihat DeleteRolePolicydi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus kebijakan inline **`S3AccessPolicy`** yang disematkan dalam peran. IAM **`S3BackupRole`**

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Untuk API detailnya, lihat DeleteRolePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteSAMLProvider`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteSAMLProvider`.

CLI

AWS CLI

Untuk menghapus SAML penyedia

Contoh ini menghapus penyedia IAM SAML 2.0 ARN yang`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`.

```
aws iam delete-saml-provider \
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia IAM SAML identitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat D eleteSAMLProvider di Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk API detailnya, lihat D eleteSAMLProvider di AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus penyedia IAM SAML 2.0 ARN yang**`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**.

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Untuk API detailnya, lihat D eleteSAMLProvider di AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteServerCertificate`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteServerCertificate`.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
   Aws::IAM::IAMClient iam(clientConfig);
   Aws::IAM::Model::DeleteServerCertificateRequest request;
   request.SetServerCertificateName(certificateName);

   const auto outcome = iam.DeleteServerCertificate(request);
   bool result = true;
   if (!outcome.IsSuccess()) {
       if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
           std::cerr << "Error deleting server certificate " << certificateName
 <<
                     ": " << outcome.GetError().GetMessage() << std::endl;
           result = false;
       }
       else {
           std::cout << "Certificate '" << certificateName
                     << "' not found." << std::endl;
```

```
        }
    }
    else {
        std::cout << "Successfully deleted server certificate " <<
 certificateName
                  << std::endl;
    }

    return result;
}
```

- Untuk API detailnya, lihat DeleteServerCertificatedi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk menghapus sertifikat server dari AWS akun Anda

`delete-server-certificate`Perintah berikut menghapus sertifikat server yang ditentukan dari AWS akun Anda.

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Perintah ini tidak menghasilkan output.

Untuk membuat daftar sertifikat server yang tersedia di AWS akun Anda, gunakan `list-server-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat server IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Hapus sertifikat server.

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat DeleteServerCertificatedi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
```

```
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat DeleteServerCertificatedi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus sertifikat server bernama**`MyServerCert`**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Untuk API detailnya, lihat DeleteServerCertificatedi AWS Tools for PowerShell Referensi Cmdlet.

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
```

```
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat DeleteServerCertificatedi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteServiceLinkedRole`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteServiceLinkedRole`.

CLI

AWS CLI

Untuk menghapus peran terkait layanan

`delete-service-linked-role`Contoh berikut menghapus peran terkait layanan tertentu yang tidak lagi Anda perlukan. Penghapusan terjadi secara asinkron. Anda dapat memeriksa status penghapusan dan mengonfirmasi kapan dilakukan dengan menggunakan perintah. `get-service-linked-role-deletion-status`

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

Output:

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat DeleteServiceLinkedRoledi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

- Untuk API detailnya, lihat DeleteServiceLinkedRoledi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Untuk API detailnya, lihat DeleteServiceLinkedRoledi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus peran terkait layanan. Harap dicatat bahwa jika layanan masih menggunakan peran ini, maka perintah ini mengakibatkan kegagalan.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Untuk API detailnya, lihat DeleteServiceLinkedRoledi AWS Tools for PowerShell Referensi Cmdlet.

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Deletes a service-linked role.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_service_linked_role(role_name)
```

```
    response = @iam_client.delete_service_linked_role(role_name: role_name)
    task_id = response.deletion_task_id
    check_deletion_status(role_name, task_id)
  rescue Aws::Errors::ServiceError => e
    handle_deletion_error(e, role_name)
  end

  private

  # Checks the deletion status of a service-linked role
  #
  # @param role_name [String] The name of the role being deleted
  # @param task_id [String] The task ID for the deletion process
  def check_deletion_status(role_name, task_id)
    loop do
      response = @iam_client.get_service_linked_role_deletion_status(
        deletion_task_id: task_id
      )
      status = response.status
      @logger.info("Deletion of #{role_name} #{status}.")
      break if %w[SUCCEEDED FAILED].include?(status)

      sleep(3)
    end
  end

  # Handles deletion error
  #
  # @param e [Aws::Errors::ServiceError] The error encountered during deletion
  # @param role_name [String] The name of the role attempted to delete
  def handle_deletion_error(e, role_name)
    return if e.code == 'NoSuchEntity'

    @logger.error("Couldn't delete #{role_name}. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk API detailnya, lihat DeleteServiceLinkedRoledi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- Untuk API detailnya, lihat DeleteServiceLinkedRole AWSSDKuntuk API referensi Rust.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteSigningCertificate`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteSigningCertificate`.

CLI

AWS CLI

Untuk menghapus sertifikat penandatanganan untuk IAM pengguna

`delete-signing-certificate`Perintah berikut menghapus sertifikat penandatanganan yang ditentukan untuk IAM pengguna bernama`Bob`.

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

Perintah ini tidak menghasilkan output.

Untuk mendapatkan ID untuk sertifikat penandatanganan, gunakan `list-signing-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat penandatanganan di Panduan EC2 Pengguna Amazon.

- Untuk API detailnya, lihat DeleteSigningCertificatedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus sertifikat penandatanganan dengan ID **`Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU`** dari IAM pengguna bernama**`Bob`**.

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Untuk API detailnya, lihat DeleteSigningCertificatedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteUser`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteUser`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat grup dan tambahkan pengguna
- Buat pengguna read-only dan read-write

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeleteUserdi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
##############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
```

```
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Untuk API detailnya, lihat DeleteUserdi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
```

```
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- Untuk API detailnya, lihat DeleteUserdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk menghapus IAM pengguna

`delete-user`Perintah berikut menghapus nama IAM pengguna Bob dari akun saat ini.

```
aws iam delete-user \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menghapus IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteUserdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
```

```
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- Untuk API detailnya, lihat DeleteUserdi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

```
}
```

- Untuk API detailnya, lihat DeleteUserdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Hapus pengguna.

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat DeleteUserdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat DeleteUserdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteIAMUser(userNameVal: String) {
```

```
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Untuk API detailnya, lihat DeleteUser AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus nama IAM **Bob** pengguna.

```
Remove-IAMUser -UserName Bob
```

Contoh 2: Contoh ini menghapus nama IAM pengguna **Theresa** bersama dengan elemen apa pun yang harus dihapus terlebih dahulu.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }
```

```
# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Untuk API detailnya, lihat DeleteUserdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
```

```
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise
```

- Untuk API detailnya, lihat DeleteUser AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
```

```
  end
```

- Untuk API detailnya, lihat DeleteUserdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
```

```
}
```

- Untuk API detailnya, lihat DeleteUser AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func deleteUser(user: IAMClientTypes.User) async throws {
        let input = DeleteUserInput(
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUser(input: input)
        } catch {
            print("ERROR: deleteUser:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat DeleteUser AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteUserPermissionsBoundary`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteUserPermissionsBoundary`.

CLI

AWS CLI

Untuk menghapus batas izin dari pengguna IAM

`delete-user-permissions-boundary`Contoh berikut menghapus batas izin yang dilampirkan ke pengguna bernama. IAM `intern` Untuk menerapkan batas izin ke pengguna, gunakan perintah. `put-user-permissions-boundary`

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteUserPermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke pengguna. IAM

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Untuk API detailnya, lihat DeleteUserPermissionsBoundarydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteUserPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DeleteUserPolicydi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menghapus kebijakan dari IAM pengguna

`delete-user-policy`Perintah berikut menghapus kebijakan yang ditentukan dari nama IAM penggunaBob.

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk mendapatkan daftar kebijakan bagi IAM pengguna, gunakan `list-user-policies` perintah.

Untuk informasi selengkapnya, lihat Membuat IAM pengguna di AWS akun Anda di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DeleteUserPolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
```

```
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Untuk API detailnya, lihat DeleteUserPolicydi AWS SDK for Go APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus kebijakan inline bernama **`AccessToEC2Policy`** yang disematkan dalam nama IAM pengguna. **`Bob`**

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Contoh 2: Contoh ini menemukan semua kebijakan inline yang disematkan dalam nama IAM pengguna **`Theresa`** dan kemudian menghapusnya.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Untuk API detailnya, lihat DeleteUserPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Deletes a user and their associated resources
  #
  # @param user_name [String] The name of the user to delete
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
```

- Untuk API detailnya, lihat DeleteUserPolicydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
```

```
        .await?;

    Ok(())
}
```

- Untuk API detailnya, lihat DeleteUserPolicy AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
 throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            print("ERROR: deleteUserPolicy:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat DeleteUserPolicy AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DeleteVirtualMfaDevice`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteVirtualMfaDevice`.

CLI

AWS CLI

Untuk menghapus MFA perangkat virtual

`delete-virtual-mfa-device`Perintah berikut menghapus MFA perangkat yang ditentukan dari akun saat ini.

```
aws iam delete-virtual-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menonaktifkan MFA perangkat di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat DeleteVirtualMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus MFA perangkat IAM virtual ARN yang**`arn:aws:iam::123456789012:mfa/bob`**.

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Contoh 2: Contoh ini memeriksa untuk melihat apakah IAM pengguna Theresa memiliki MFA perangkat yang ditetapkan. Jika ditemukan, perangkat dinonaktifkan untuk IAM pengguna. Jika perangkat virtual, maka itu juga dihapus.

```
$mfa = Get-IAMMFADevice -UserName Theresa
```

```
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Untuk API detailnya, lihat DeleteVirtualMfaDevicedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DetachGroupPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DetachGroupPolicy`.

CLI

AWS CLI

Untuk melepaskan kebijakan dari grup

Contoh ini menghapus kebijakan terkelola dengan ARN `arn:aws:iam::123456789012:policy/TesterAccessPolicy` dari grup yang dipanggil`Testers`.

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengelola grup IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DetachGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola ARN yang **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** berasal dari grup bernama**`Testers`**.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada grup bernama **`Testers`** dan memisahkannya dari grup.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Untuk API detailnya, lihat DetachGroupPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DetachRolePolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DetachRolePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Kelola peran

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat DetachRolePolicydi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat DetachRolePolicydi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
 "
                  << roleName << std::endl;
    }

    return detachOutcome.IsSuccess();
```

- Untuk API detailnya, lihat DetachRolePolicydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk melepaskan kebijakan dari peran

Contoh ini menghapus kebijakan terkelola dengan ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` dari peran yang dipanggil`FedTesterRole`.

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DetachRolePolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}
```

- Untuk API detailnya, lihat DetachRolePolicydi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
```

```
 */
public class DetachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }

    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

        } catch (IamException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat DetachRolePolicydi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Lepaskan kebijakan.

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.

- Untuk API detailnya, lihat DetachRolePolicydi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
```

```
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat DetachRolePolicydi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun detachPolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        DetachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

- Untuk API detailnya, lihat DetachRolePolicy AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola ARN yang **`arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy`** berasal dari peran bernama**`FedTesterRole`**.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada peran yang diberi nama **`FedTesterRole`** dan memisahkannya dari peran.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Untuk API detailnya, lihat DetachRolePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Lepaskan kebijakan dari peran menggunakan objek Kebijakan Boto3.

```
def detach_from_role(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
```

```
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

Lepaskan kebijakan dari peran menggunakan objek Peran Boto3.

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- Untuk API detailnya, lihat DetachRolePolicy AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat DetachRolePolicydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
```

```
}
```

- Untuk API detailnya, lihat DetachRolePolicy AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
 IAMClientTypes.Role) async throws {
        let input = DetachRolePolicyInput(
            policyArn: policy.arn,
            roleName: role.roleName
        )

        do {
            _ = try await iamClient.detachRolePolicy(input: input)
        } catch {
            print("ERROR: detachRolePolicy:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat DetachRolePolicy AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DetachUserPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DetachUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

CLI

AWS CLI

Untuk melepaskan kebijakan dari pengguna

Contoh ini menghapus kebijakan terkelola dengan ARN `arn:aws:iam::123456789012:policy/TesterPolicy` dari penggunaBob.

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengubah izin untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DetachUserPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini melepaskan kebijakan terkelola ARN yang **`arn:aws:iam::123456789012:policy/TesterPolicy`** berasal dari nama IAM **`Bob`** pengguna.

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada nama IAM pengguna **`Theresa`** dan melepaskan kebijakan tersebut dari pengguna.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Untuk API detailnya, lihat DetachUserPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

- Untuk API detailnya, lihat DetachUserPolicy AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Detaches a policy from a user
 #
 # @param user_name [String] The name of the user
 # @param policy_arn [String] The ARN of the policy to detach
 # @return [Boolean] true if the policy was successfully detached, false
otherwise
 def detach_user_policy(user_name, policy_arn)
   @iam_client.detach_user_policy(
     user_name: user_name,
     policy_arn: policy_arn
   )
   @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
   true
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error('Error detaching policy: Policy or user does not exist.')
   false
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error detaching policy from user '#{user_name}':
#{e.message}")
   false
 end
```

- Untuk API detailnya, lihat DetachUserPolicydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Untuk API detailnya, lihat DetachUserPolicy AWSSDKuntuk API referensi Rust.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`EnableMfaDevice`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`EnableMfaDevice`.

CLI

AWS CLI

Untuk mengaktifkan MFA perangkat

Setelah Anda menggunakan `create-virtual-mfa-device` perintah untuk membuat MFA perangkat virtual baru, Anda dapat menetapkan MFA perangkat ke pengguna. `enable-mfa-device`Contoh berikut menetapkan MFA perangkat dengan nomor seri `arn:aws:iam::210987654321:mfa/BobsMFADevice` kepada penggunaBob. Perintah ini juga menyinkronkan perangkat AWS dengan memasukkan dua kode pertama secara berurutan dari MFA perangkat virtual.

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengaktifkan perangkat autentikasi multi-faktor virtual (MFA) di Panduan Pengguna.AWS IAM

- Untuk API detailnya, lihat EnableMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini memungkinkan MFA perangkat keras dengan nomor seri **987654321098** dan mengaitkan perangkat dengan pengguna**Bob**. Ini termasuk dua kode pertama secara berurutan dari perangkat.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Contoh 2: Contoh ini membuat dan mengaktifkan MFA perangkat virtual. Perintah pertama membuat perangkat virtual dan mengembalikan representasi objek perangkat dalam variabel**$MFADevice**. Anda dapat menggunakan **QRCodePng** properti **.Base32StringSeed** atau untuk mengkonfigurasi aplikasi perangkat lunak pengguna. Perintah terakhir menetapkan perangkat kepada pengguna**David**, mengidentifikasi perangkat dengan nomor serinya. Perintah ini juga menyinkronkan perangkat AWS dengan memasukkan dua kode pertama secara berurutan dari MFA perangkat virtual.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
```

```
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Untuk API detailnya, lihat EnableMfaDevicedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GenerateCredentialReport`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GenerateCredentialReport`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk menghasilkan laporan kredensi

Contoh berikut mencoba untuk menghasilkan laporan kredensi untuk AWS akun.

```
aws iam generate-credential-report
```

Output:

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

Untuk informasi selengkapnya, lihat Mendapatkan laporan kredensi untuk AWS akun Anda di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GenerateCredentialReportdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini meminta pembuatan laporan baru, yang dapat dilakukan setiap empat jam. Jika laporan terakhir masih terbaru, bidang Negara berbunyi**COMPLETE**. Gunakan **Get-IAMCredentialReport** untuk melihat laporan yang sudah selesai.

```
Request-IAMCredentialReport
```

Output:

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- Untuk API detailnya, lihat GenerateCredentialReportdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
```

```
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
%s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
account.")
        raise
    else:
        return response
```

- Untuk API detailnya, lihat GenerateCredentialReport AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GenerateServiceLastAccessedDetails`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GenerateServiceLastAccessedDetails`.

CLI

AWS CLI

Contoh 1: Untuk membuat laporan akses layanan untuk kebijakan kustom

`generate-service-last-accessed-details`Contoh berikut memulai pekerjaan latar belakang untuk menghasilkan laporan yang mencantumkan layanan yang diakses oleh IAM pengguna dan entitas lain dengan kebijakan kustom bernama`intern-boundary`. Anda dapat menampilkan laporan setelah dibuat dengan menjalankan `get-service-last-accessed-details` perintah.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

Output:

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

Contoh 2: Untuk membuat laporan akses layanan untuk AdministratorAccess kebijakan AWS terkelola

`generate-service-last-accessed-details`Contoh berikut memulai pekerjaan latar belakang untuk menghasilkan laporan yang mencantumkan layanan yang diakses oleh IAM pengguna dan entitas lain dengan `AdministratorAccess` kebijakan AWS terkelola. Anda dapat menampilkan laporan setelah dibuat dengan menjalankan `get-service-last-accessed-details` perintah.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

Output:

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

Untuk informasi selengkapnya, lihat Menyempurnakan izin dalam AWS menggunakan informasi yang terakhir diakses di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat GenerateServiceLastAccessedDetailsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini setara dengan cmdlet. GenerateServiceLastAccessedDetails API Ini menyediakan id pekerjaan yang dapat digunakan di Get- IAMServiceLastAccessedDetail dan Get- IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Untuk API detailnya, lihat GenerateServiceLastAccessedDetailsdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetAccessKeyLastUsed`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccessKeyLastUsed`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kunci akses

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);

    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
```

```
                    secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat GetAccessKeyLastUseddi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan

Contoh berikut mengambil informasi tentang kapan kunci akses ABCDEXAMPLE terakhir digunakan.

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

Output:

```
{
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

Untuk informasi selengkapnya, lihat Mengelola kunci akses untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetAccessKeyLastUseddi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan kunci akses.

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);

  if (response.AccessKeyLastUsed?.LastUsedDate) {
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
    the ${response.AccessKeyLastUsed.ServiceName} service on
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat GetAccessKeyLastUseddi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
  }
);
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat GetAccessKeyLastUseddi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan nama pengguna yang dimiliki dan informasi penggunaan terakhir untuk kunci akses yang disediakan.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Untuk API detailnya, lihat GetAccessKeyLastUseddi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
```

```
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

- Untuk API detailnya, lihat GetAccessKeyLastUsed AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetAccountAuthorizationDetails`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccountAuthorizationDetails`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk membuat daftar AWS akun, IAM pengguna, grup, peran, dan kebijakan

`get-account-authorization-details`Perintah berikut menampilkan informasi tentang semua IAM pengguna, grup, peran, dan kebijakan di AWS akun.

```
aws iam get-account-authorization-details
```

Output:

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
```

```
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "RoleId": "AROA1234567890EXAMPLE",
            "CreateDate": "2014-07-30T17:09:20Z",
            "InstanceProfileList": [
                {
                    "InstanceProfileId": "AIPA1234567890EXAMPLE",
                    "Roles": [
                        {
                            "AssumeRolePolicyDocument": {
                                "Version": "2012-10-17",
                                "Statement": [
                                    {
                                        "Sid": "",
                                        "Effect": "Allow",
                                        "Principal": {
                                            "Service": "ec2.amazonaws.com"
                                        },
                                        "Action": "sts:AssumeRole"
                                    }
                                ]
                            },
                            "RoleId": "AROA1234567890EXAMPLE",
                            "CreateDate": "2014-07-30T17:09:20Z",
                            "RoleName": "EC2role",
                            "Path": "/",
                            "Arn": "arn:aws:iam::123456789012:role/EC2role"
                        }
                    ],
                    "CreateDate": "2014-07-30T17:09:20Z",
                    "InstanceProfileName": "EC2role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
                }
            ],
```

```
            "RoleName": "EC2role",
            "Path": "/",
            "AttachedManagedPolicies": [
                {
                    "PolicyName": "AmazonS3FullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
                },
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
```

```
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
                                "Effect": "Allow"
                            }
                        ]
                    }
                }
            ]
        }
    ],
    "UserDetailList": [
        {
            "UserName": "Alice",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:24Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Alice"
        },
        {
            "UserName": "Bob",
            "GroupList": [
                "Admins"
```

```
            ],
            "CreateDate": "2013-10-14T18:32:25Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [
                {
                    "PolicyName": "DenyBillingAndIAMPolicy",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": {
                            "Effect": "Deny",
                            "Action": [
                                "aws-portal:*",
                                "iam:*"
                            ],
                            "Resource": "*"
                        }
                    }
                }
            ],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        },
        {
            "UserName": "Charlie",
            "GroupList": [
                "Dev"
            ],
            "CreateDate": "2013-10-14T18:33:56Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Charlie"
        }
    ],
    "Policies": [
        {
            "PolicyName": "create-update-delete-set-managed-policies",
            "CreateDate": "2015-02-06T19:58:34Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
```

```
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-02-06T19:58:34Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": {
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": [
```

```
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
                                    "arn:aws:s3:::example-bucket",
                                    "arn:aws:s3:::example-bucket/*"
                                ]
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
            "UpdateDate": "2015-01-21T23:39:41Z"
        },
        {
            "PolicyName": "AmazonEC2FullAccess",
            "CreateDate": "2015-02-06T18:40:15Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2014-10-30T20:59:46Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Action": "ec2:*",
                                "Effect": "Allow",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "elasticloadbalancing:*",
                                "Resource": "*"
```

```
                            },
                            {
                                "Effect": "Allow",
                                "Action": "cloudwatch:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

Untuk informasi selengkapnya, lihat pedoman audit AWS keamanan di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetAccountAuthorizationDetailsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mendapatkan rincian otorisasi tentang identitas di AWS akun, dan menampilkan daftar elemen objek yang dikembalikan, termasuk pengguna, grup, dan peran. Misalnya, **`UserDetailList`** properti menampilkan detail tentang pengguna. Informasi serupa tersedia di **`RoleDetailList`** dan **`GroupDetailList`** properti.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Output:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Output:

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Untuk API detailnya, lihat GetAccountAuthorizationDetailsdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- Untuk API detailnya, lihat GetAccountAuthorizationDetails AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetAccountPasswordPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccountPasswordPolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- Untuk API detailnya, lihat GetAccountPasswordPolicydi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk melihat kebijakan kata sandi akun saat ini

`get-account-password-policy`Perintah berikut menampilkan detail tentang kebijakan kata sandi untuk akun saat ini.

```
aws iam get-account-password-policy
```

Output:

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

Jika tidak ada kebijakan kata sandi yang ditentukan untuk akun, perintah mengembalikan `NoSuchEntity` kesalahan.

Untuk informasi selengkapnya, lihat Menyetel kebijakan kata sandi akun untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetAccountPasswordPolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}
```

```
// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- Untuk API detailnya, lihat GetAccountPasswordPolicydi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan kebijakan kata sandi akun.

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
  const command = new GetAccountPasswordPolicyCommand({});
```

```
  const response = await client.send(command);
  console.log(response.PasswordPolicy);
  return response;
};
```

- Untuk API detailnya, lihat GetAccountPasswordPolicydi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Untuk API detailnya, lihat GetAccountPasswordPolicydi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan detail tentang kebijakan kata sandi untuk akun saat ini. Jika tidak ada kebijakan kata sandi yang ditentukan untuk akun, perintah mengembalikan **`NoSuchEntity`** kesalahan.

```
Get-IAMAccountPasswordPolicy
```

Output:

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Untuk API detailnya, lihat GetAccountPasswordPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def print_password_policy():
    """
    Prints the password policy for the account.
    """
    try:
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
 {pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
```

```
        print(f"\tpassword_reuse_prevention:
 {pw_policy.password_reuse_prevention}")
        print(
            f"\trequire_lowercase_characters:
 {pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
 {pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- Untuk API detailnya, lihat [GetAccountPasswordPolicy AWS](SDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS].

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger
```

```
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'IAMPolicyManager'
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    response = @iam_client.get_account_password_policy
    @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.info('The account does not have a password policy.')
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't print the account password policy. Error: #{e.code} -
 #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat GetAccountPasswordPolicydi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- Untuk API detailnya, lihat GetAccountPasswordPolicy AWSSDKuntuk API referensi Rust.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetAccountSummary`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccountSummary`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk mendapatkan informasi tentang penggunaan IAM entitas dan IAM kuota di akun saat ini

`get-account-summary`Perintah berikut mengembalikan informasi tentang penggunaan IAM entitas saat ini dan kuota IAM entitas saat ini di akun.

```
aws iam get-account-summary
```

Output:

```
{
    "SummaryMap": {
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
```

```
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

Untuk informasi selengkapnya tentang batasan entitas, lihat IAMdan AWS STS kuota di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetAccountSummarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan informasi tentang penggunaan IAM entitas saat ini dan kuota IAM entitas saat ini di Akun AWS.

```
Get-IAMAccountSummary
```

Output:

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
Groups                                  3
```

```
UsersQuota                          5000
RolePolicySizeQuota                 10240
UserPolicySizeQuota                 2048
GroupsPerUserQuota                  10
AssumeRolePolicySizeQuota           2048
AttachedPoliciesPerGroupQuota       2
Roles                               9
VersionsPerPolicyQuota              5
GroupsQuota                         100
PolicySizeQuota                     5120
Policies                            5
RolesQuota                          250
ServerCertificates                  0
AttachedPoliciesPerRoleQuota        2
MFADevicesInUse                     2
PoliciesQuota                       1000
AccountMFAEnabled                   1
Providers                           2
InstanceProfilesQuota               100
MFADevices                          4
AccessKeysPerUserQuota              2
AttachedPoliciesPerUserQuota        2
SigningCertificatesPerUserQuota     2
PolicyVersionsInUse                 4
InstanceProfiles                    1
...
```

- Untuk API detailnya, lihat GetAccountSummarydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_summary():
```

```
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- Untuk API detailnya, lihat GetAccountSummary AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetContextKeysForCustomPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetContextKeysForCustomPolicy`.

CLI

AWS CLI

Contoh 1: Untuk mencantumkan kunci konteks yang direferensikan oleh satu atau beberapa JSON kebijakan khusus yang disediakan sebagai parameter pada baris perintah

`get-context-keys-for-custom-policy`Perintah berikut mem-parsing setiap kebijakan yang disediakan dan mencantumkan kunci konteks yang digunakan oleh kebijakan tersebut. Gunakan perintah ini untuk mengidentifikasi nilai kunci konteks mana yang harus Anda berikan agar berhasil menggunakan perintah simulator kebijakan `simulate-custom-policy` dan`simulate-custom-policy`. Anda juga dapat mengambil daftar kunci konteks yang digunakan oleh semua kebijakan yang terkait oleh IAM pengguna atau peran dengan menggunakan `get-context-keys-for-custom-policy` perintah. Nilai parameter yang

dimulai dengan `file://` menginstruksikan perintah untuk membaca file dan menggunakan konten sebagai nilai untuk parameter, bukan nama file itu sendiri.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

Output:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Contoh 2: Untuk mencantumkan kunci konteks yang direferensikan oleh satu atau beberapa JSON kebijakan kustom yang disediakan sebagai input file

`get-context-keys-for-custom-policy`Perintah berikut ini sama dengan contoh sebelumnya, kecuali bahwa kebijakan disediakan dalam file, bukan sebagai parameter. Karena perintah mengharapkan JSON daftar string, dan bukan daftar struktur, file harus JSON terstruktur sebagai berikut, meskipun Anda dapat menciutkannya menjadi satu.

```
[
    "Policy1",
    "Policy2"
]
```

Jadi misalnya, file yang berisi kebijakan dari contoh sebelumnya harus terlihat seperti berikut. Anda harus menghindari setiap kutipan ganda yang disematkan di dalam string kebijakan dengan mendahuluinya dengan garis miring terbalik".

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow
\", \"Action\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-
west-2:128716708097:table/${aws:username}\", \"Condition\": {\"DateGreaterThan\":
 {\"aws:CurrentTime\": \"2015-08-16T12:00:00Z\"}}}}" ]
```

File ini kemudian dapat dikirimkan ke perintah berikut.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

Output:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan Simulator IAM Kebijakan (AWS CLIdan AWS API) di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetContextKeysForCustomPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam kebijakan yang disediakan JSON.Untuk memberikan beberapa kebijakan yang dapat Anda berikan sebagai daftar nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Untuk API detailnya, lihat GetContextKeysForCustomPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetContextKeysForPrincipalPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetContextKeysForPrincipalPolicy`.

CLI

AWS CLI

Untuk membuat daftar kunci konteks yang direferensikan oleh semua kebijakan yang terkait dengan prinsipal IAM

`get-context-keys-for-principal-policy`Perintah berikut mengambil semua kebijakan yang dilampirkan ke pengguna `saanvi` dan grup mana pun yang menjadi anggotanya. Kemudian mem-parsing masing-masing dan mencantumkan kunci konteks yang digunakan oleh kebijakan tersebut. Gunakan perintah ini untuk mengidentifikasi nilai kunci konteks mana yang harus Anda berikan agar berhasil menggunakan `simulate-principal-policy` perintah `simulate-custom-policy` dan. Anda juga dapat mengambil daftar kunci konteks yang digunakan oleh JSON kebijakan arbitrer dengan menggunakan perintah. `get-context-keys-for-custom-policy`

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

Output:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan Simulator IAM Kebijakan (AWS CLIdan AWS API) di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetContextKeysForPrincipalPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam json kebijakan yang disediakan dan kebijakan yang dilampirkan ke IAM entitas (pengguna/peran, dll.). Untuk -PolicyInputList Anda dapat memberikan beberapa daftar nilai sebagai nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Untuk API detailnya, lihat GetContextKeysForPrincipalPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetCredentialReport`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetCredentialReport`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk mendapatkan laporan kredensi

Contoh ini membuka laporan yang dikembalikan dan mengeluarkannya ke pipeline sebagai array baris teks.

```
aws iam get-credential-report
```

Output:

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
}
```

Untuk informasi selengkapnya, lihat Mendapatkan laporan kredensi untuk AWS akun Anda di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetCredentialReportdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuka laporan yang dikembalikan dan mengeluarkannya ke pipeline sebagai array baris teks. Baris pertama adalah header dengan nama kolom yang dipisahkan koma. Setiap baris berturut-turut adalah baris detail untuk satu pengguna, dengan setiap bidang dipisahkan dengan koma. Sebelum Anda dapat melihat laporan, Anda harus membuatnya dengan **`Request-IAMCredentialReport`** cmdlet. Untuk mengambil laporan sebagai string tunggal, gunakan **`-Raw`** sebagai pengganti. **`-AsTextArray`** Alias juga **`-SplitLines`** diterima untuk **`-AsTextArray`** sakelar. Untuk daftar lengkap kolom dalam output, lihat API referensi layanan. Perhatikan bahwa jika Anda tidak menggunakan **`-AsTextArray`** atau**`-SplitLines`**, maka Anda harus mengekstrak teks dari **`.Content`** properti menggunakan. NET**`StreamReader`**kelas.

```
Request-IAMCredentialReport
```

Output:

```
Description                                                    State
-----------                                                    -----
```

```
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Output:

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Untuk API detailnya, lihat GetCredentialReportdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
```

```
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- Untuk API detailnya, lihat GetCredentialReport AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetGroup`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetGroup`.

CLI

AWS CLI

Untuk mendapatkan IAM grup

Contoh ini mengembalikan rincian tentang IAM grup`Admins`.

```
aws iam get-group \
    --group-name Admins
```

Output:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    },
    "Users": []
```

```
}
```

Untuk informasi selengkapnya, lihat IAMIdentitas (pengguna, grup pengguna, dan peran) di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menampilkan detail tentang IAM grup**`Testers`**, termasuk kumpulan semua IAM pengguna yang termasuk dalam grup.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Output:

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Untuk API detailnya, lihat GetGroupdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetGroupPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetGroupPolicy`.

CLI

AWS CLI

Untuk mendapatkan informasi tentang kebijakan yang dilampirkan ke IAM grup

`get-group-policy`Perintah berikut mendapatkan informasi tentang kebijakan tertentu yang dilampirkan ke grup bernama`Test-Group`.

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

Output:

```
{
```

```
    "GroupName": "Test-Group",
    "PolicyDocument": {
        "Statement": [
            {
                "Action": [
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

Untuk informasi selengkapnya, lihat Mengelola IAM kebijakan di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan inline tertanam yang dinamai **`PowerUserAccess-Testers`** untuk grup**`Testers`**. **`PolicyDocument`**Properti URL dikodekan. Itu diterjemahkan dalam contoh ini dengan. **`UrlDecode`** NETmetode.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Output:

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "NotAction": "iam:*",
      "Resource": "*"
    }
  ]
}
```

- Untuk API detailnya, lihat GetGroupPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetInstanceProfile`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetInstanceProfile`.

CLI

AWS CLI

Untuk mendapatkan informasi tentang profil instance

`get-instance-profile`Perintah berikut mendapatkan informasi tentang profil instance bernama`ExampleInstanceProfile`.

```
aws iam get-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Output:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
```

```
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/
ExampleInstanceProfile"
    }
}
```

Untuk informasi selengkapnya, lihat Menggunakan profil instans di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetInstanceProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan rincian profil contoh bernama **`ec2instancerole`** yang didefinisikan dalam AWS akun saat ini.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Untuk API detailnya, lihat GetInstanceProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetLoginProfile`** dengan a CLI

Contoh kode berikut menunjukkan cara menggunakan`GetLoginProfile`.

CLI

AWS CLI

Untuk mendapatkan informasi kata sandi untuk IAM pengguna

`get-login-profile`Perintah berikut mendapatkan informasi tentang password untuk IAM pengguna bernama`Bob`.

```
aws iam get-login-profile \
    --user-name Bob
```

Output:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

`get-login-profile`Perintah tersebut dapat digunakan untuk memverifikasi bahwa IAM pengguna memiliki kata sandi. Perintah mengembalikan `NoSuchEntity` kesalahan jika tidak ada kata sandi yang ditentukan untuk pengguna.

Anda tidak dapat melihat kata sandi menggunakan perintah ini. Jika kata sandi hilang, Anda dapat mengatur ulang kata sandi (`update-login-profile`) untuk pengguna. Atau, Anda dapat menghapus profil login (`delete-login-profile`) untuk pengguna dan kemudian membuat yang baru (`create-login-profile`).

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan tanggal pembuatan kata sandi dan apakah reset kata sandi diperlukan untuk IAM pengguna**David**.

```
Get-IAMLoginProfile -UserName David
```

Output:

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
12/10/2014 3:39:44 PM      False                              David
```

- Untuk API detailnya, lihat GetLoginProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetOpenIdConnectProvider`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetOpenIdConnectProvider`.

CLI

AWS CLI

Untuk mengembalikan informasi tentang penyedia OpenID Connect yang ditentukan

Contoh ini mengembalikan rincian tentang penyedia OpenID Connect yangARN. `arn:aws:iam::123456789012:oidc-provider/server.example.com`

```
aws iam get-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Output:

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat GetOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan rincian tentang penyedia OpenID Connect yangARN. **`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`** **`ClientIDList`**Properti adalah koleksi yang berisi semua Klien yang IDs ditentukan untuk penyedia ini.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Output:

```
ClientIDList         CreateDate               ThumbprintList
           Url
------------         ----------               --------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Untuk API detailnya, lihat GetOpenIdConnectProviderdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Bekerja dengan Pembuat IAM Kebijakan API

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }
```

- Untuk API detailnya, lihat GetPolicydi AWS SDK for .NET APIReferensi.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat GetPolicydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mengambil informasi tentang kebijakan terkelola yang ditentukan

Contoh ini mengembalikan rincian tentang kebijakan terkelola ARN yang`arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetPolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Untuk API detailnya, lihat GetPolicydi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan kebijakan.

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat GetPolicydi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
};

iam.getPolicy(params, function (err, data) {
  if (err) {
```

```
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat GetPolicydi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun getIAMPolicy(policyArnVal: String?) {
    val request =
        GetPolicyRequest {
            policyArn = policyArnVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Untuk API detailnya, lihat GetPolicy AWSSDKAPIreferensi Kotlin.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
    {
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Untuk API detailnya, lihat GetPolicydi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan terkelola ARN yang**`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
```

```
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Untuk API detailnya, lihat GetPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- Untuk API detailnya, lihat GetPolicy AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end
```

- Untuk API detailnya, lihat GetPolicydi AWS SDK for Ruby APIReferensi.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
        let input = GetPolicyInput(
            policyArn: arn
        )
        do {
            let output = try await client.getPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: getPolicy:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat GetPolicy AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetPolicyVersion`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetPolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Bekerja dengan Pembuat IAM Kebijakan API

CLI

AWS CLI

Untuk mengambil informasi tentang versi tertentu dari kebijakan terkelola yang ditentukan

Contoh ini mengembalikan dokumen kebijakan untuk versi v2 dari kebijakan ARN yang`arn:aws:iam::123456789012:policy/MyManagedPolicy`.

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Output:

```
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetPolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan dokumen kebijakan untuk **v2** versi kebijakan ARN yang**`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Dokumen kebijakan

di **`Document`** properti URL dikodekan dan diterjemahkan dalam contoh ini dengan file. **`UrlDecode`** NETmetode.

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Output:

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version": "2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
}
```

- Untuk API detailnya, lihat GetPolicyVersiondi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- Untuk API detailnya, lihat GetPolicyVersion AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetRole`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }
```

- Untuk API detailnya, lihat GetRoledi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk mendapatkan informasi tentang IAM peran

`get-role`Perintah berikut mendapatkan informasi tentang peran bernama`Test-Role`.

```
aws iam get-role \
    --role-name Test-Role
```

Output:

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

Perintah menampilkan kebijakan kepercayaan yang dilampirkan pada peran tersebut. Untuk mencantumkan kebijakan izin yang dilampirkan ke peran, gunakan `list-role-policies` perintah.

Untuk informasi selengkapnya, lihat Membuat IAM peran di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetRoledi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
```

```
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Untuk API detailnya, lihat GetRoledi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan perannya.

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
```

```
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Untuk API detailnya, lihat GetRoledi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Untuk API detailnya, lihat GetRoledi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan rincian**lamda_exec_role**. Ini termasuk dokumen kebijakan kepercayaan yang menentukan siapa yang dapat mengambil peran ini. Dokumen kebijakan URL dikodekan dan dapat diterjemahkan menggunakan file. NET**UrlDecode**metode. Dalam contoh ini, kebijakan asli menghapus semua spasi putih sebelum diunggah ke kebijakan. Untuk melihat dokumen kebijakan izin yang menentukan apa yang dapat dilakukan oleh seseorang yang mengasumsikan peran tersebut, gunakan kebijakan **Get-IAMRolePolicy** for inline, dan **Get-IAMPolicyVersion** untuk kebijakan terkelola terlampir.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Output:

```
Arn                      : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate               : 4/2/2015 9:16:11 AM
Path                     : /
RoleId                   : 2YBIKAIBHNKB4EXAMPLE1
RoleName                 : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Output:

```
{"Version":"2012-10-17","Statement":[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Untuk API detailnya, lihat GetRoledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_role(role_name):
    """
    Gets a role by name.

    :param role_name: The name of the role to retrieve.
    :return: The specified role.
    """
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- Untuk API detailnya, lihat GetRole AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Gets data about a role.
  #
  # @param name [String] The name of the role to look up.
  # @return [Aws::IAM::Role] The retrieved role.
  def get_role(name)
    role = @iam_client.get_role({
                                  role_name: name
                                }).role
    puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't get data for role '#{name}' Here's why:")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end
```

- Untuk API detailnya, lihat GetRoledi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- Untuk API detailnya, lihat GetRole AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func getRole(name: String) async throws -> IAMClientTypes.Role {
        let input = GetRoleInput(
            roleName: name
        )
        do {
            let output = try await client.getRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            print("ERROR: getRole:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat GetRole AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetRolePolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetRolePolicy`.

CLI

AWS CLI

Untuk mendapatkan informasi tentang kebijakan yang dilampirkan pada IAM peran

`get-role-policy`Perintah berikut mendapatkan informasi tentang kebijakan tertentu yang dilampirkan pada peran bernama`Test-Role`.

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Output:

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
      "Statement": [
          {
              "Action": [
                  "s3:ListBucket",
                  "s3:Put*",
                  "s3:Get*",
                  "s3:*MultipartUpload*"
              ],
              "Resource": "*",
              "Effect": "Allow",
              "Sid": "1"
          }
      ]
  }
  "PolicyName": "ExamplePolicy"
}
```

Untuk informasi selengkapnya, lihat Membuat IAM peran di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetRolePolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan dokumen kebijakan izin untuk kebijakan bernama **`oneClick_lambda_exec_role_policy`** yang disematkan dalam IAM peran**`lamda_exec_role`**. Dokumen kebijakan yang dihasilkan URL dikodekan. Itu diterjemahkan dalam contoh ini dengan. **`UrlDecode`** NETmetode.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Output:

```
PolicyDocument                                                  PolicyName
              UserName
--------------                                                  ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Output:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
```

```
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    }
  ]
}
```

- Untuk API detailnya, lihat GetRolePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetSamlProvider`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetSamlProvider`.

CLI

AWS CLI

Untuk mengambil metadokumen SAML penyedia

Contoh ini mengambil rincian tentang penyedia SAML 2.0 ARM yang`arn:aws:iam::123456789012:saml-provider/SAMLADFS`. Responsnya mencakup dokumen metadata yang Anda dapatkan dari penyedia identitas untuk membuat entitas AWS SAML penyedia serta tanggal pembuatan dan kedaluwarsa.

```
aws iam get-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Output:

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
```

```
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia IAM SAML identitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetSamlProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil rincian tentang penyedia SAML 2.0 yang arn:aws:iam: ARM :123456789012:saml-provider/. SAMLADFS Responsnya mencakup dokumen metadata yang Anda dapatkan dari penyedia identitas untuk membuat entitas AWS SAML penyedia serta tanggal pembuatan dan kedaluwarsa.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Output:

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Untuk API detailnya, lihat GetSamlProviderdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetServerCertificate`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServerCertificate`.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
```

```
                    std::endl;
    }

    return result;
}
```

- Untuk API detailnya, lihat GetServerCertificatedi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mendapatkan detail tentang sertifikat server di AWS akun Anda

`get-server-certificate`Perintah berikut mengambil semua detail tentang sertifikat server yang ditentukan di AWS akun Anda.

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Output:

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
```

```
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```

Untuk membuat daftar sertifikat server yang tersedia di AWS akun Anda, gunakan `list-server-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat server IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan sertifikat server.

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat GetServerCertificatedi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getServerCertificate(
```

```
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat GetServerCertificatedi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil rincian tentang sertifikat server bernama**MyServerCertificate**. Anda dapat menemukan rincian sertifikat di **CertificateBody** dan **ServerCertificateMetadata** properti.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Output:

```
CertificateBody           : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
```

```
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                            21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk API detailnya, lihat GetServerCertificatedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetServiceLastAccessedDetails`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServiceLastAccessedDetails`.

CLI

AWS CLI

Untuk mengambil laporan akses layanan

`get-service-last-accessed-details`Contoh berikut mengambil laporan yang dibuat sebelumnya yang mencantumkan layanan yang diakses oleh IAM entitas. Untuk menghasilkan laporan, gunakan `generate-service-last-accessed-details` perintah.

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

Output:

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

Untuk informasi selengkapnya, lihat Menyempurnakan izin dalam AWS menggunakan informasi yang terakhir diakses di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat GetServiceLastAccessedDetailsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memberikan rincian layanan yang terakhir diakses oleh IAM entitas (pengguna, grup, peran, atau kebijakan) yang terkait dalam panggilan Permintaan.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Output:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Untuk API detailnya, lihat GetServiceLastAccessedDetailsdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetServiceLastAccessedDetailsWithEntities`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServiceLastAccessedDetailsWithEntities`.

CLI

AWS CLI

Untuk mengambil laporan akses layanan dengan rincian untuk layanan

`get-service-last-accessed-details-with-entities`Contoh berikut mengambil laporan yang berisi rincian tentang IAM pengguna dan entitas lain yang mengakses layanan tertentu. Untuk menghasilkan laporan, gunakan `generate-service-last-accessed-details` perintah. Untuk mendapatkan daftar layanan yang diakses dengan ruang nama, gunakan. `get-service-last-accessed-details`

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

Output:

```
{
    "JobStatus": "COMPLETED",
```

```
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Menyempurnakan izin dalam AWS menggunakan informasi yang terakhir diakses di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat GetServiceLastAccessedDetailsWithEntitiesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memberikan stempel waktu terakhir yang diakses untuk layanan dalam permintaan oleh entitas masing-masingIAM.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Output:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Output:

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Output:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Untuk API detailnya, lihat GetServiceLastAccessedDetailsWithEntitiesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetServiceLinkedRoleDeletionStatus`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServiceLinkedRoleDeletionStatus`.

CLI

AWS CLI

Untuk memeriksa status permintaan untuk menghapus peran terkait layanan

`get-service-linked-role-deletion-status`Contoh berikut menampilkan status permintaan sebelumnya untuk menghapus peran terkait layanan. Operasi hapus terjadi secara asinkron. Ketika Anda membuat permintaan, Anda mendapatkan `DeletionTaskId` nilai yang Anda berikan sebagai parameter untuk perintah ini.

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Output:

```
{
"Status": "SUCCEEDED"
}
```

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat GetServiceLinkedRoleDeletionStatusdi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
```

```
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

  return client.send(command);
};
```

- Untuk API detailnya, lihat GetServiceLinkedRoleDeletionStatusdi AWS SDK for JavaScript APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetUser`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetUser`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Get information about an IAM user.
```

```
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
 { UserName = userName });
        return response.User;
    }
```

- Untuk API detailnya, lihat GetUserdi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
```

```
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- Untuk API detailnya, lihat GetUserdi Referensi AWS CLI Perintah.

CLI

AWS CLI

Untuk mendapatkan informasi tentang IAM pengguna

`get-user`Perintah berikut mendapatkan informasi tentang nama IAM penggunaPaulo.

```
aws iam get-user \
    --user-name Paulo
```

Output:

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

Untuk informasi selengkapnya, lihat Mengelola IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetUserdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
```

```
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}
```

- Untuk API detailnya, lihat GetUserdi AWS SDK for Go APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil rincian tentang nama **David** pengguna.

```
Get-IAMUser -UserName David
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Contoh 2: Contoh ini mengambil detail tentang pengguna yang saat ini masuk. IAM

```
Get-IAMUser
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Untuk API detailnya, lihat GetUserdi AWS Tools for PowerShell Referensi Cmdlet.

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Retrieves a user's details
 #
 # @param user_name [String] The name of the user to retrieve
 # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
 def get_user(user_name)
   response = @iam_client.get_user(user_name: user_name)
   response.user
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("User '#{user_name}' not found.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error retrieving user '#{user_name}': #{e.message}")
   nil
 end
```

- Untuk API detailnya, lihat GetUserdi AWS SDK for Ruby APIReferensi.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func getUser(name: String? = nil) async throws -> IAMClientTypes.User
 {
        let input = GetUserInput(
            userName: name
        )
        do {
            let output = try await iamClient.getUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            return user
        } catch {
            print("ERROR: getUser:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat GetUser AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetUserPolicy`** dengan a CLI

Contoh kode berikut menunjukkan cara menggunakan`GetUserPolicy`.

CLI

AWS CLI

Untuk mencantumkan detail kebijakan untuk IAM pengguna

`get-user-policy`Perintah berikut mencantumkan rincian kebijakan tertentu yang dilampirkan ke nama IAM penggunaBob.

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Output:

```
{
    "UserName": "Bob",
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    }
}
```

Untuk mendapatkan daftar kebijakan bagi IAM pengguna, gunakan `list-user-policies` perintah.

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat GetUserPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil rincian kebijakan inline bernama **`Davids_IAM_Admin_Policy`** yang disematkan dalam nama IAM pengguna. **`David`** Dokumen kebijakan URL dikodekan.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Output:

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

- Untuk API detailnya, lihat GetUserPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListAccessKeys`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAccessKeys`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kunci akses

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
# Returns:
```

```
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
 the specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }

  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
    --output text \
```

```
    --query 'AccessKeyMetadata[].AccessKeyId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat ListAccessKeysdi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
                      << ": " << outcome.GetError().GetMessage() << std::endl;
```

```
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
                      std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                      statusString << std::setw(20) <<
                      key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk API detailnya, lihat ListAccessKeysdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk membuat daftar kunci akses IDs untuk IAM pengguna

`list-access-keys`Perintah berikut mencantumkan kunci akses IDs untuk IAM pengguna bernama`Bob`.

```
aws iam list-access-keys \
    --user-name Bob
```

Output:

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

Anda tidak dapat mencantumkan kunci akses rahasia untuk IAM pengguna. Jika kunci akses rahasia hilang, Anda harus membuat kunci akses baru menggunakan `create-access-keys` perintah.

Untuk informasi selengkapnya, lihat Mengelola kunci akses untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListAccessKeysdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Untuk API detailnya, lihat ListAccessKeysdi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccessKeys {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user for which access keys are
 retrieved.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void listKeys(IamClient iam, String userName) {
        try {
            boolean done = false;
            String newMarker = null;

            while (!done) {
                ListAccessKeysResponse response;

                if (newMarker == null) {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .build();

                    response = iam.listAccessKeys(request);

                } else {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .marker(newMarker)
                            .build();

                    response = iam.listAccessKeys(request);
                }

                for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                    System.out.format("Retrieved access key %s",
metadata.accessKeyId());
                }

                if (!response.isTruncated()) {
```

```
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat ListAccessKeysdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kunci akses.

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
```

```
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);

  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat ListAccessKeysdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
```

```
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat ListAccessKeysdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listKeys(userNameVal: String?) {
    val request =
        ListAccessKeysRequest {
            userName = userNameVal
        }
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
```

```
        }
    }
}
```

- Untuk API detailnya, lihat ListAccessKeys AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mencantumkan kunci akses untuk IAM pengguna bernama**Bob**. Perhatikan bahwa Anda tidak dapat mencantumkan kunci akses rahasia untuk IAM pengguna. Jika kunci akses rahasia hilang, Anda harus membuat kunci akses baru dengan **`New-IAMAccessKey`** cmdlet.

```
Get-IAMAccessKey -UserName "Bob"
```

Output:

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Untuk API detailnya, lihat ListAccessKeysdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys
```

- Untuk API detailnya, lihat ListAccessKeys AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end
```

```
  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
```

```
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat ListAccessKeysdi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListAccountAliases`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAccountAliases`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }

        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
```

```
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk API detailnya, lihat ListAccountAliasesdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk membuat daftar alias akun

`list-account-aliases`Perintah berikut mencantumkan alias untuk akun saat ini.

```
aws iam list-account-aliases
```

Output:

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

Untuk informasi selengkapnya, lihat ID AWS akun Anda dan aliasnya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListAccountAliasesdi Referensi AWS CLI Perintah.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccountAliases {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAliases(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat ListAccountAliasesdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar alias akun.

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
      yield alias;
    }
```

```
    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat ListAccountAliasesdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat ListAccountAliasesdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listAliases() {
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- Untuk API detailnya, lihat ListAccountAliases AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan alias akun untuk. Akun AWS

```
Get-IAMAccountAlias
```

Output:

```
ExampleCo
```

- Untuk API detailnya, lihat ListAccountAliasesdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

- Untuk API detailnya, lihat ListAccountAliases AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat ListAccountAliasesdi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListAttachedGroupPolicies`** dengan a CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAttachedGroupPolicies`.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke grup yang ditentukan

Contoh ini mengembalikan nama dan kebijakan ARNs terkelola yang dilampirkan ke IAM grup bernama `Admins` di AWS akun.

```
aws iam list-attached-group-policies \
    --group-name Admins
```

Output:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListAttachedGroupPoliciesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan nama dan ARNs kebijakan terkelola yang dilampirkan ke IAM grup bernama **`Admins`** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam grup, gunakan **`Get-IAMGroupPolicyList`** perintah.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Output:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- Untuk API detailnya, lihat ListAttachedGroupPoliciesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListAttachedRolePolicies`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAttachedRolePolicies`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
   /// <summary>
   /// List the IAM role policies that are attached to an IAM role.
   /// </summary>
   /// <param name="roleName">The IAM role to list IAM policies for.</param>
   /// <returns>A list of the IAM policies attached to the IAM role.</returns>
   public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
   {
       var attachedPolicies = new List<AttachedPolicyType>();
       var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

       await foreach (var response in attachedRolePoliciesPaginator.Responses)
       {
           attachedPolicies.AddRange(response.AttachedPolicies);
       }

       return attachedPolicies;
   }
```

- Untuk API detailnya, lihat ListAttachedRolePoliciesdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan terkelola yang dilampirkan pada peran yang ditentukan

Perintah ini mengembalikan nama dan kebijakan ARNs terkelola yang dilampirkan pada IAM peran yang disebutkan `SecurityAuditRole` di AWS akun.

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

Output:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListAttachedRolePoliciesdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
```

```
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}
```

- Untuk API detailnya, lihat ListAttachedRolePoliciesdi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kebijakan yang dilampirkan pada peran.

```
import {
  ListAttachedRolePoliciesCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });

  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk API detailnya, lihat ListAttachedRolePoliciesdi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- Untuk API detailnya, lihat ListAttachedRolePoliciesdi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan nama dan kebijakan ARNs terkelola yang dilampirkan pada IAM peran yang disebutkan **`SecurityAuditRole`** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam peran, gunakan **`Get-IAMRolePolicyList`** perintah.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Output:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- Untuk API detailnya, lihat ListAttachedRolePoliciesdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
            logger.info("Got policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- Untuk API detailnya, lihat ListAttachedRolePolicies AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat ListAttachedRolePoliciesdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_attached_role_policies(
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
        .set_path_prefix(path_prefix)
        .set_marker(marker)
```

```
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Untuk API detailnya, lihat ListAttachedRolePolicies AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3



    /// Returns a list of AWS Identity and Access Management (IAM) policies
    /// that are attached to the role.
    ///
    /// - Parameter role: The IAM role to return the policy list for.
    ///
    /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
    ///   describing each managed policy that's attached to the role.
    public func listAttachedRolePolicies(role: String) async throws ->
 [IAMClientTypes.AttachedPolicy] {
        var policyList: [IAMClientTypes.AttachedPolicy] = []

        // Use "Paginated" to get all the attached role polices.
        // This lets the SDK handle the 'isTruncated' in
 "ListAttachedRolePoliciesOutput".
        let input = ListAttachedRolePoliciesInput(
            roleName: role
        )
```

```
        let output = client.listAttachedRolePoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let attachedPolicies = page.attachedPolicies else {
                    print("Error: no attached policies returned.")
                    continue
                }
                for attachedPolicy in attachedPolicies {
                    policyList.append(attachedPolicy)
                }
            }
        } catch {
            print("ERROR: listAttachedRolePolicies:", dump(error))
            throw error
        }

        return policyList
    }
```

- Untuk API detailnya, lihat ListAttachedRolePolicies AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListAttachedUserPolicies`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAttachedUserPolicies`.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke pengguna yang ditentukan

Perintah ini mengembalikan nama dan ARNs kebijakan terkelola untuk IAM pengguna yang disebutkan Bob di AWS akun.

```
aws iam list-attached-user-policies \
```

```
    --user-name Bob
```

Output:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListAttachedUserPoliciesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan nama dan ARNs kebijakan terkelola untuk IAM pengguna yang disebutkan **Bob** di AWS akun. Untuk melihat daftar kebijakan inline yang disematkan di IAM pengguna, gunakan **Get-IAMUserPolicyList** perintah.

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Output:

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/TesterPolicy                     TesterPolicy
```

- Untuk API detailnya, lihat ListAttachedUserPoliciesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListEntitiesForPolicy`** dengan a CLI

Contoh kode berikut menunjukkan cara menggunakan`ListEntitiesForPolicy`.

CLI

AWS CLI

Untuk mencantumkan semua pengguna, grup, dan peran yang dilampirkan oleh kebijakan terkelola yang ditentukan

Contoh ini menampilkan daftar IAM grup, peran, dan pengguna yang memiliki kebijakan yang `arn:aws:iam::123456789012:policy/TestPolicy` dilampirkan.

```
aws iam list-entities-for-policy \
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

Output:

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
            "UserName": "Alice",
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
```

```
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListEntitiesForPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menampilkan daftar IAM grup, peran, dan pengguna yang memiliki kebijakan yang **`arn:aws:iam::123456789012:policy/TestPolicy`** dilampirkan.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Output:

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Untuk API detailnya, lihat ListEntitiesForPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListGroupPolicies`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListGroupPolicies`.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan inline yang dilampirkan ke grup yang ditentukan

`list-group-policies`Perintah berikut mencantumkan nama-nama kebijakan inline yang dilampirkan ke IAM grup bernama `Admins` di akun saat ini.

```
aws iam list-group-policies \
    --group-name Admins
```

Output:

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola IAM kebijakan di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListGroupPoliciesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan daftar kebijakan inline yang disematkan dalam grup**`Testers`**. Untuk mendapatkan kebijakan terkelola yang dilampirkan ke grup, gunakan perintah**`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Output:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Untuk API detailnya, lihat ListGroupPoliciesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListGroups`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListGroups`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- Untuk API detailnya, lihat ListGroupsdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk membuat daftar IAM grup untuk akun saat ini

`list-groups`Perintah berikut mencantumkan IAM grup di akun saat ini.

```
aws iam list-groups
```

Output:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola grup IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListGroupsdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(ctx context.Context, maxGroups int32)
 ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(ctx, &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
 return groups, err
}
```

- Untuk API detailnya, lihat ListGroupsdi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar grup.

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk API detailnya, lihat ListGroupsdi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listGroupsArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listGroups($listGroupsArguments);
    }
```

- Untuk API detailnya, lihat ListGroupsdi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan koleksi semua IAM kelompok didefinisikan dalam saat ini Akun AWS.

```
Get-IAMGroupList
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Untuk API detailnya, lihat ListGroupsdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_groups(count):
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
```

```
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- Untuk API detailnya, lihat ListGroups AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

```
end
```

- Untuk API detailnya, lihat ListGroupsdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Untuk API detailnya, lihat ListGroups AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listGroups() async throws -> [String] {
        var groupList: [String] = []

        // Use "Paginated" to get all the groups.
        // This lets the SDK handle the 'isTruncated' property in
 "ListGroupsOutput".
        let input = ListGroupsInput()

        let pages = client.listGroupsPaginated(input: input)
        do {
            for try await page in pages {
                guard let groups = page.groups else {
                    print("Error: no groups returned.")
                    continue
                }

                for group in groups {
                    if let name = group.groupName {
                        groupList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listGroups:", dump(error))
            throw error
        }
        return groupList
    }
```

- Untuk API detailnya, lihat ListGroups AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListGroupsForUser`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListGroupsForUser`.

CLI

AWS CLI

Untuk membuat daftar grup yang dimiliki IAM pengguna

`list-groups-for-user`Perintah berikut menampilkan grup yang Bob menjadi milik IAM pengguna.

```
aws iam list-groups-for-user \
    --user-name Bob
```

Output:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
```

```
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola grup IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListGroupsForUserdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan daftar IAM grup yang **David** dimiliki IAM pengguna.

```
Get-IAMGroupForUser -UserName David
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Untuk API detailnya, lihat ListGroupsForUserdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListInstanceProfiles`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListInstanceProfiles`.

CLI

AWS CLI

Untuk mencantumkan profil instance untuk akun

`list-instance-profiles`Perintah berikut mencantumkan profil instance yang terkait dengan akun saat ini.

```
aws iam list-instance-profiles
```

Output:

```
{
    "InstanceProfiles": [
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
```

```
                    }
                }
            ]
        },
        {
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan profil instans di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListInstanceProfilesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan koleksi profil instance didefinisikan dalam saat ini Akun AWS.

```
Get-IAMInstanceProfileList
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Untuk API detailnya, lihat ListInstanceProfilesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListInstanceProfilesForRole`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListInstanceProfilesForRole`.

CLI

AWS CLI

Untuk membuat daftar profil instance untuk IAM peran

`list-instance-profiles-for-role`Perintah berikut mencantumkan profil instance yang terkait dengan peran tersebut`Test-Role`.

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

Output:

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan profil instans di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListInstanceProfilesForRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan rincian profil instance yang terkait dengan peran**`ec2instancerole`**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Output:

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Untuk API detailnya, lihat ListInstanceProfilesForRoledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListMfaDevices`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListMfaDevices`.

CLI

AWS CLI

Untuk mencantumkan semua MFA perangkat untuk pengguna tertentu

Contoh ini mengembalikan detail tentang MFA perangkat yang ditetapkan ke IAM penggunaBob.

```
aws iam list-mfa-devices \
    --user-name Bob
```

Output:

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
        },
        {
```

```
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multi-faktor (MFA) AWS di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat ListMfaDevicesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan detail tentang MFA perangkat yang ditetapkan ke IAM pengguna**David**. Dalam contoh ini Anda dapat mengatakan bahwa itu adalah perangkat virtual karena ARN bukan nomor seri aktual perangkat fisik. **SerialNumber**

```
Get-IAMMFADevice -UserName David
```

Output:

```
EnableDate                SerialNumber                            UserName
----------                ------------                            --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David     David
```

- Untuk API detailnya, lihat ListMfaDevicesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListOpenIdConnectProviders`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListOpenIdConnectProviders`.

CLI

AWS CLI

Untuk mencantumkan informasi tentang penyedia OpenID Connect di akun AWS

Contoh ini mengembalikan daftar ARNS semua penyedia OpenID Connect yang didefinisikan dalam akun saat ini AWS .

```
aws iam list-open-id-connect-providers
```

Output:

```
{
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat ListOpenIdConnectProvidersdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan daftar ARNS semua penyedia OpenID Connect yang didefinisikan dalam saat ini. Akun AWS

```
Get-IAMOpenIDConnectProviderList
```

Output:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Untuk API detailnya, lihat ListOpenIdConnectProvidersdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListPolicies`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListPolicies`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
```

```
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for .NET APIReferensi.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
```

```
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(64) << "Description" << std::setw(12) <<
                      "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mencantumkan kebijakan terkelola yang tersedia untuk AWS akun Anda

Contoh ini mengembalikan kumpulan dari dua kebijakan terkelola pertama yang tersedia di AWS akun saat ini.

```
aws iam list-policies \
    --max-items 3
```

Output:

```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
```

```
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListPoliciesdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
```

```
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kebijakan.

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
    Scope: "Local",
  });

  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
```

```
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
```

```
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan kumpulan dari tiga kebijakan terkelola pertama yang tersedia di AWS akun saat ini. Karena tidak **`-scope`** ditentukan, defaultnya **`all`** dan mencakup kebijakan terkelola dan yang AWS dikelola pelanggan.

```
Get-IAMPolicyList -MaxItem 3
```

Output:

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
```

```
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Contoh 2: Contoh ini mengembalikan kumpulan dari dua kebijakan terkelola pelanggan pertama yang tersedia di AWS akun saat ini. Ini digunakan **`-Scope local`** untuk membatasi output hanya pada kebijakan yang dikelola pelanggan.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Output:

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
```

```
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Untuk API detailnya, lihat ListPoliciesdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- Untuk API detailnya, lihat ListPolicies AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat ListPoliciesdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;
```

```
    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Untuk API detailnya, lihat ListPolicies AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listPolicies() async throws -> [MyPolicyRecord] {
        var policyList: [MyPolicyRecord] = []

        // Use "Paginated" to get all the policies.
        // This lets the SDK handle the 'isTruncated' in "ListPoliciesOutput".
        let input = ListPoliciesInput()
        let output = client.listPoliciesPaginated(input: input)

        do {
            for try await page in output {
```

```
                guard let policies = page.policies else {
                    print("Error: no policies returned.")
                    continue
                }

                for policy in policies {
                    guard let name = policy.policyName,
                          let id = policy.policyId,
                          let arn = policy.arn
                    else {
                        throw ServiceHandlerError.noSuchPolicy
                    }
                    policyList.append(MyPolicyRecord(name: name, id: id, arn:
arn))
                }
            }
        } catch {
            print("ERROR: listPolicies:", dump(error))
            throw error
        }

        return policyList
    }
```

- Untuk API detailnya, lihat ListPolicies AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListPolicyVersions`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListPolicyVersions`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Kembalikan versi kebijakan

CLI

AWS CLI

Untuk mencantumkan informasi tentang versi kebijakan terkelola yang ditentukan

Contoh ini menampilkan daftar versi kebijakan yang ARN tersedia`arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
{
    "IsTruncated": false,
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListPolicyVersionsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan daftar versi kebijakan yang ARN tersedia**`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Untuk mendapatkan dokumen kebijakan untuk versi tertentu, gunakan **`Get-IAMPolicyVersion`** perintah dan tentukan **`VersionId`** yang Anda inginkan.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Output:

```
CreateDate                    Document                IsDefaultVersion
       VersionId
----------                    --------                ---------------
       ---------
2/12/2015 9:39:53 AM                                  True
       v2
2/12/2015 9:39:09 AM                                  False
       v1
```

- Untuk API detailnya, lihat ListPolicyVersionsdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListRolePolicies`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListRolePolicies`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
```

```
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
_IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- Untuk API detailnya, lihat ListRolePoliciesdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk membuat daftar kebijakan yang dilampirkan pada IAM peran

`list-role-policies`Perintah berikut mencantumkan nama kebijakan izin untuk IAM peran yang ditentukan.

```
aws iam list-role-policies \
    --role-name Test-Role
```

Output:

```
{
    "PolicyNames": [
        "ExamplePolicy"
    ]
}
```

Untuk melihat kebijakan kepercayaan yang dilampirkan pada peran, gunakan `get-role` perintah. Untuk melihat detail kebijakan izin, gunakan `get-role-policy` perintah.

Untuk informasi selengkapnya, lihat Membuat IAM peran di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListRolePoliciesdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Untuk API detailnya, lihat ListRolePoliciesdi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kebijakan.

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.PolicyNames?.length) {
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
```

```
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk API detailnya, lihat ListRolePoliciesdi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
        if ($marker) {
            $listRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

```
```

- Untuk API detailnya, lihat ListRolePoliciesdi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan daftar nama kebijakan inline yang disematkan dalam IAM peran**lamda_exec_role**. Untuk melihat detail kebijakan inline, gunakan perintah**Get-IAMRolePolicy**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Output:

```
oneClick_lambda_exec_role_policy
```

- Untuk API detailnya, lihat ListRolePoliciesdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_policies(role_name):
    """
    Lists inline policies for a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
```

```
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
        raise
```

- Untuk API detailnya, lihat ListRolePolicies AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

- Untuk API detailnya, lihat ListRolePoliciesdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Untuk API detailnya, lihat ListRolePolicies AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listRolePolicies(role: String) async throws -> [String] {
        var policyList: [String] = []

        // Use "Paginated" to get all the role policies.
        // This lets the SDK handle the 'isTruncated' in
 "ListRolePoliciesOutput".
        let input = ListRolePoliciesInput(
            roleName: role
        )
        let pages = client.listRolePoliciesPaginated(input: input)

        do {
            for try await page in pages {
                guard let policies = page.policyNames else {
                    print("Error: no role policies returned.")
                    continue
                }

                for policy in policies {
                    policyList.append(policy)
                }
            }
        } catch {
            print("ERROR: listRolePolicies:", dump(error))
            throw error
        }
        return policyList
    }
```

- Untuk API detailnya, lihat ListRolePolicies AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListRoleTags`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListRoleTags`.

CLI

AWS CLI

Untuk mencantumkan tag yang dilampirkan pada peran

`list-role-tags`Perintah berikut mengambil daftar tag yang terkait dengan peran yang ditentukan.

```
aws iam list-role-tags \
    --role-name production-role
```

Output:

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Menandai IAM sumber daya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListRoleTagsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil tag yang terkait dengan peran..

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Untuk API detailnya, lihat ListRoleTagsdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListRoles`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListRoles`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

- Untuk API detailnya, lihat ListRolesdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk membuat daftar IAM peran untuk akun saat ini

`list-roles`Perintah berikut mencantumkan IAM peran untuk akun saat ini.

```
aws iam list-roles
```

Output:

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
```

```
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat IAM peran di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListRolesdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

- Untuk API detailnya, lihat ListRolesdi AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar peran.

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
```

```
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk API detailnya, lihat ListRolesdi AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listRolesArguments["MaxItems"] = $maxItems;
        }
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- Untuk API detailnya, lihat ListRolesdi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil daftar semua IAM peran dalam. Akun AWS

```
Get-IAMRoleList
```

- Untuk API detailnya, lihat ListRolesdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.

    :param count: The number of roles to list.
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- Untuk API detailnya, lihat ListRoles AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
# Lists IAM roles up to a specified count.
# @param count [Integer] the maximum number of roles to list.
# @return [Array<String>] the names of the roles.
def list_roles(count)
  role_names = []
  roles_counted = 0

  @iam_client.list_roles.each_page do |page|
    page.roles.each do |role|
      break if roles_counted >= count

      @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
      role_names << role.role_name
      roles_counted += 1
    end
    break if roles_counted >= count
  end

  role_names
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't list roles for the account. Here's why:")
  @logger.error("\t#{e.code}: #{e.message}")
  raise
end
```

- Untuk API detailnya, lihat ListRolesdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Untuk API detailnya, lihat ListRoles AWSSDKAPIreferensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
```

```
import AWSS3


    public func listRoles() async throws -> [String] {
        var roleList: [String] = []

        // Use "Paginated" to get all the roles.
        // This lets the SDK handle the 'isTruncated' in "ListRolesOutput".
        let input = ListRolesInput()
        let pages = client.listRolesPaginated(input: input)

        do {
            for try await page in pages {
                guard let roles = page.roles else {
                    print("Error: no roles returned.")
                    continue
                }

                for role in roles {
                    if let name = role.roleName {
                        roleList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listRoles:", dump(error))
            throw error
        }
        return roleList
    }
```

- Untuk API detailnya, lihat ListRoles AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListSAMLProviders`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListSAMLProviders`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- Untuk API detailnya, lihat L istSAMLProviders di AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk daftar SAML penyedia di AWS akun

Contoh ini mengambil daftar penyedia SAML 2.0 yang dibuat di AWS akun saat ini.

```
aws iam list-saml-providers
```

Output:

```
{
    "SAMLProviderList": [
        {
```

```
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia IAM SAML identitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat L istSAMLProviders di Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
```

```
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- Untuk API detailnya, lihat L istSAMLProviders di AWS SDK for Go APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Daftar SAML penyedia.

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk API detailnya, lihat L istSAMLProviders di AWS SDK for JavaScript APIReferensi.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Untuk API detailnya, lihat L istSAMLProviders di AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil daftar penyedia SAML 2.0 yang dibuat saat ini Akun AWS. Ini mengembalikan, tanggal pembuatanARN, dan tanggal kedaluwarsa untuk setiap SAML penyedia.

```
Get-IAMSAMLProviderList
```

Output:

```
Arn                                                 CreateDate
   ValidUntil
---                                                 ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS    12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Untuk API detailnya, lihat L istSAMLProviders di AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
    try:
        found = 0
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- Untuk API detailnya, lihat L AWS SDKuntuk istSAMLProviders Referensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat L istSAMLProviders di AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Untuk API detailnya, lihat L istSAMLProviders AWS SDKuntuk API referensi Rust.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListServerCertificates`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListServerCertificates`.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listServerCertificates(
```

```
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();

        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
                      certificate.GetArn() << std::setw(14) <<

certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::setw(14) <<

certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
```

```
            done = true;
        }
    }

    return true;
}
```

- Untuk API detailnya, lihat ListServerCertificatesdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mencantumkan sertifikat server di AWS akun Anda

`list-server-certificates`Perintah berikut mencantumkan semua sertifikat server yang disimpan dan tersedia untuk digunakan di AWS akun Anda.

```
aws iam list-server-certificates
```

Output:

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
```

```
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola sertifikat server IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListServerCertificatesdi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar sertifikat.

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }

    if (response.IsTruncated) {
```

```
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat ListServerCertificatesdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat ListServerCertificatesdi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil daftar sertifikat server yang telah diunggah ke saat ini. Akun AWS

```
Get-IAMServerCertificateList
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk API detailnya, lihat ListServerCertificatesdi AWS Tools for PowerShell Referensi Cmdlet.

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end
```

```
  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat ListServerCertificatesdi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListSigningCertificates`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListSigningCertificates`.

CLI

AWS CLI

Untuk membuat daftar sertifikat penandatanganan untuk IAM pengguna

`list-signing-certificates`Perintah berikut mencantumkan sertifikat penandatanganan untuk IAM pengguna bernamaBob.

```
aws iam list-signing-certificates \
    --user-name Bob
```

Output:

```
{
    "Certificates": [
```

```
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-
body>-----END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola sertifikat penandatanganan di Panduan EC2 Pengguna Amazon.

- Untuk API detailnya, lihat ListSigningCertificatesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil rincian tentang sertifikat penandatanganan yang terkait dengan nama **Bob** pengguna.

```
Get-IAMSigningCertificate -UserName Bob
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
```

```
                    YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                    21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                    rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                    nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                    NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                    -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Untuk API detailnya, lihat ListSigningCertificatesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListUserPolicies`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListUserPolicies`.

CLI

AWS CLI

Untuk membuat daftar kebijakan untuk IAM pengguna

`list-user-policies`Perintah berikut mencantumkan kebijakan yang dilampirkan ke nama IAM penggunaBob.

```
aws iam list-user-policies \
    --user-name Bob
```

Output:

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat IAM pengguna di AWS akun Anda di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListUserPoliciesdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Untuk API detailnya, lihat ListUserPoliciesdi AWS SDK for Go APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil daftar nama kebijakan inline yang disematkan dalam nama IAM pengguna. **`David`**

```
Get-IAMUserPolicyList -UserName David
```

Output:

```
Davids_IAM_Admin_Policy
```

- Untuk API detailnya, lihat ListUserPoliciesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListUserTags`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListUserTags`.

CLI

AWS CLI

Untuk mencantumkan tag yang dilampirkan ke pengguna

`list-user-tags`Perintah berikut mengambil daftar tag yang terkait dengan IAM pengguna tertentu.

```
aws iam list-user-tags \
    --user-name alice
```

Output:

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Menandai IAM sumber daya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListUserTagsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil tag yang terkait dengan pengguna.

```
Get-IAMUserTagList -UserName joe
```

- Untuk API detailnya, lihat ListUserTagsdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListUsers`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListUsers`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Untuk API detailnya, lihat ListUsersdi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
```

```
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat ListUsersdi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }
```

```
        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk API detailnya, lihat ListUsersdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk daftar IAM pengguna

`list-users`Perintah berikut mencantumkan IAM pengguna di akun saat ini.

```
aws iam list-users
```

Output:

```
{
    "Users": [
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        }
    ]
```

```
}
```

Untuk informasi selengkapnya, lihat Daftar IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat ListUsersdi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}
```

- Untuk API detailnya, lihat ListUsersdi AWS SDK for Go APIReferensi.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAllUsers(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
```

```
        try {
            boolean done = false;
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
 ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }

                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
 user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

 permissionsBoundary.permissionsBoundaryTypeAsString());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat ListUsersdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Daftar pengguna.

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);

  for (const { UserName, CreateDate } of response.Users) {
    console.log(`${UserName} created on: ${CreateDate}`);
  }
  return response;
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat ListUsersdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
```

```
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat ListUsersdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listAllUsers() {
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
```

```
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null) {
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
            }
        }
    }
}
```

- Untuk API detailnya, lihat ListUsers AWSSDKAPIreferensi Kotlin.

PHP

SDKuntuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- Untuk API detailnya, lihat ListUsersdi AWS SDK for PHP APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil koleksi pengguna saat ini Akun AWS.

```
Get-IAMUserList
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Administrator
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE1
UserName         : Administrator

Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/6/2015 12:54:42 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : L3EWNONDOM3YUEXAMPLE2
UserName         : bab

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Untuk API detailnya, lihat ListUsersdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- Untuk API detailnya, lihat ListUsers AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Lists all users in the AWS account
  #
  # @return [Array<Aws::IAM::Types::User>] An array of user objects
  def list_users
    users = []
    @iam_client.list_users.each_page do |page|
      page.users.each do |user|
        users << user
      end
    end
    users
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing users: #{e.message}")
    []
  end
```

- Untuk API detailnya, lihat ListUsersdi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_users(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
```

```
        .await?;
    Ok(response)
}
```

- Untuk API detailnya, lihat ListUsers AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listUsers() async throws -> [MyUserRecord] {
        var userList: [MyUserRecord] = []

        // Use "Paginated" to get all the users.
        // This lets the SDK handle the 'isTruncated' in "ListUsersOutput".
        let input = ListUsersInput()
        let output = client.listUsersPaginated(input: input)

        do {
            for try await page in output {
                guard let users = page.users else {
                    continue
                }
                for user in users {
                    if let id = user.userId, let name = user.userName {
                        userList.append(MyUserRecord(id: id, name: name))
                    }
                }
            }
        }
        catch {
```

```
            print("ERROR: listUsers:", dump(error))
            throw error
        }
       return userList
    }
```

- Untuk API detailnya, lihat ListUsers AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ListVirtualMfaDevices`** dengan a CLI

Contoh kode berikut menunjukkan cara menggunakan`ListVirtualMfaDevices`.

CLI

AWS CLI

Untuk daftar MFA perangkat virtual

`list-virtual-mfa-devices`Perintah berikut mencantumkan MFA perangkat virtual yang telah dikonfigurasi untuk akun saat ini.

```
aws iam list-virtual-mfa-devices
```

Output:

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengaktifkan perangkat autentikasi multi-faktor virtual (MFA) di Panduan Pengguna.AWS IAM

- Untuk API detailnya, lihat ListVirtualMfaDevicesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil koleksi MFA perangkat virtual yang ditetapkan untuk pengguna di AWS akun. **`User`**Properti masing-masing adalah objek dengan detail IAM pengguna tempat perangkat ditugaskan.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Output:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Untuk API detailnya, lihat ListVirtualMfaDevicesdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`PutGroupPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutGroupPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat grup dan tambahkan pengguna

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat PutGroupPolicydi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menambahkan kebijakan ke grup

`put-group-policy`Perintah berikut menambahkan kebijakan ke IAM grup bernama`Admins`.

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

Perintah ini tidak menghasilkan output.

Kebijakan didefinisikan sebagai JSON dokumen dalam AdminPolicyfile.json. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk informasi selengkapnya, lihat Mengelola IAM kebijakan di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat PutGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat kebijakan inline bernama **`AppTesterPolicy`** dan menyematkannya dalam grup. IAM **`AppTesters`** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten JSON kebijakan datang file**`apptesterpolicy.json`**. Perhatikan bahwa Anda harus menggunakan **`-Raw`** parameter untuk berhasil memproses konten JSON file.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Untuk API detailnya, lihat PutGroupPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`PutRolePermissionsBoundary`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`PutRolePermissionsBoundary`.

CLI

AWS CLI

Contoh 1: Untuk menerapkan batas izin berdasarkan kebijakan kustom ke peran IAM

`put-role-permissions-boundary`Contoh berikut menerapkan kebijakan kustom yang dinamai `intern-boundary` sebagai batas izin untuk peran yang ditentukan. IAM

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --role-name lambda-application-role
```

Perintah ini tidak menghasilkan output.

Contoh 2: Untuk menerapkan batas izin berdasarkan kebijakan AWS terkelola ke peran IAM

`put-role-permissions-boundary`Contoh berikut menerapkan `PowerUserAccess` kebijakan AWS terkelola sebagai batas izin untuk peran yang ditentukan. IAM

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat PutRolePermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk Peran. IAM Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk API detailnya, lihat PutRolePermissionsBoundarydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`PutRolePolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutRolePolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
```

```
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat PutRolePolicydi AWS SDK for .NET APIReferensi.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(policyDocument);

    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }
```

```
    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat PutRolePolicydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk melampirkan kebijakan izin ke peran IAM

`put-role-policy`Perintah berikut menambahkan kebijakan izin ke peran bernama`Test-Role`.

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Perintah ini tidak menghasilkan output.

Kebijakan didefinisikan sebagai JSON dokumen dalam AdminPolicyfile.json. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk melampirkan kebijakan kepercayaan ke peran, gunakan `update-assume-role-policy` perintah.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat PutRolePolicydi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::some-test-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
    },
  ],
});

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
```

```
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk API detailnya, lihat PutRolePolicydi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat kebijakan inline bernama **FedTesterRolePolicy** dan menyematkannya dalam peran. IAM **FedTesterRole** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten JSON kebijakan berasal dari file**FedTesterPolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten JSON file.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Untuk API detailnya, lihat PutRolePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **PutUserPermissionsBoundary** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`PutUserPermissionsBoundary`.

CLI

AWS CLI

Contoh 1: Untuk menerapkan batas izin berdasarkan kebijakan kustom untuk pengguna IAM

`put-user-permissions-boundary`Contoh berikut menerapkan kebijakan kustom bernama `intern-boundary` sebagai batas izin untuk pengguna tertentu. IAM

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

Perintah ini tidak menghasilkan output.

Contoh 2: Untuk menerapkan batas izin berdasarkan kebijakan AWS terkelola ke pengguna IAM

`put-user-permissions-boundary`Contoh berikut menerapkan pollicy AWS terkelola bernama `PowerUserAccess` sebagai batas izin untuk pengguna tertentu. IAM

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin IAM identitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat PutUserPermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk pengguna. Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk API detailnya, lihat PutUserPermissionsBoundarydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`PutUserPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya

CLI

AWS CLI

Untuk melampirkan kebijakan ke IAM pengguna

`put-user-policy`Perintah berikut melampirkan kebijakan ke IAM pengguna bernama`Bob`.

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Perintah ini tidak menghasilkan output.

Kebijakan didefinisikan sebagai JSON dokumen dalam AdminPolicyfile.json. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin IAM identitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat PutUserPolicydi Referensi AWS CLI Perintah.

Go

SDKuntuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
```

```
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Untuk API detailnya, lihat PutUserPolicydi AWS SDK for Go APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat kebijakan inline bernama **EC2AccessPolicy** dan menyematkannya di pengguna. IAM **Bob** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten JSON kebijakan berasal dari file**EC2AccessPolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten JSON file.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Untuk API detailnya, lihat PutUserPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Creates an inline policy for a specified user.
  # @param username [String] The name of the IAM user.
  # @param policy_name [String] The name of the policy to create.
  # @param policy_document [String] The JSON policy document.
  # @return [Boolean]
  def create_user_policy(username, policy_name, policy_document)
    @iam_client.put_user_policy({
                                  user_name: username,
                                  policy_name: policy_name,
                                  policy_document: policy_document
                                })
    @logger.info("Policy #{policy_name} created for user #{username}.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    false
  end
```

- Untuk API detailnya, lihat PutUserPolicydi AWS SDK for Ruby APIReferensi.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    func putUserPolicy(policyDocument: String, policyName: String, user:
 IAMClientTypes.User) async throws {
        let input = PutUserPolicyInput(
            policyDocument: policyDocument,
```

```
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.putUserPolicy(input: input)
        } catch {
            print("ERROR: putUserPolicy:", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat PutUserPolicy AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`RemoveClientIdFromOpenIdConnectProvider`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`RemoveClientIdFromOpenIdConnectProvider`.

CLI

AWS CLI

Untuk menghapus ID klien yang ditentukan dari daftar klien yang IDs terdaftar untuk penyedia IAM OpenID Connect yang ditentukan

Contoh ini menghapus ID klien `My-TestApp-3` dari daftar klien IDs yang terkait dengan IAM OIDC penyedia ARN yang`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`.

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat RemoveClientIdFromOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus ID klien **My-TestApp-3** dari daftar klien IDs yang terkait dengan IAM OIDC penyedia ARN yang**arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**.

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Untuk API detailnya, lihat RemoveClientIdFromOpenIdConnectProviderdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **RemoveRoleFromInstanceProfile** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`RemoveRoleFromInstanceProfile`.

CLI

AWS CLI

Untuk menghapus peran dari profil instance

`remove-role-from-instance-profile`Perintah berikut menghapus peran bernama `Test-Role` dari profil instance bernama`ExampleInstanceProfile`.

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
```

```
    --role-name Test-Role
```

Untuk informasi selengkapnya, lihat Menggunakan profil instans di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat RemoveRoleFromInstanceProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus peran bernama **MyNewRole** dari profil EC2 instance bernama**MyNewRole**. Profil instance yang dibuat di IAM konsol selalu memiliki nama yang sama dengan peran, seperti dalam contoh ini. Jika Anda membuatnya di API atauCLI, maka mereka dapat memiliki nama yang berbeda.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Untuk API detailnya, lihat RemoveRoleFromInstanceProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **RemoveUserFromGroup** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`RemoveUserFromGroup`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat grup dan tambahkan pengguna

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk API detailnya, lihat RemoveUserFromGroupdi AWS SDK for .NET APIReferensi.

CLI

AWS CLI

Untuk menghapus pengguna dari IAM grup

`remove-user-from-group`Perintah berikut menghapus nama pengguna Bob dari IAM grup bernama`Admins`.

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menambahkan dan menghapus IAM pengguna di grup AWS IAMpengguna di Panduan Pengguna.

- Untuk API detailnya, lihat RemoveUserFromGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus IAM pengguna **`Bob`** dari grup**`Testers`**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Contoh 2: Contoh ini menemukan grup di mana IAM pengguna **`Theresa`** adalah anggota, dan kemudian dihapus **`Theresa`** dari grup tersebut.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Contoh 3: Contoh ini menunjukkan cara alternatif untuk menghapus IAM pengguna **`Bob`** dari **`Testers`** grup.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Untuk API detailnya, lihat RemoveUserFromGroupdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`ResyncMfaDevice`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ResyncMfaDevice`.

CLI

AWS CLI

Untuk menyinkronkan perangkat MFA

`resync-mfa-device`Contoh berikut menyinkronkan MFA perangkat yang terkait dengan IAM pengguna Bob dan yang ARN `arn:aws:iam::123456789012:mfa/BobsMFADevice` dengan program autentikator yang menyediakan dua kode otentikasi.

```
aws iam resync-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multi-faktor (MFA) AWS di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat ResyncMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menyinkronkan MFA perangkat yang terkait dengan IAM pengguna **`Bob`** dan yang ARN **`arn:aws:iam::123456789012:mfa/bob`** dengan program autentikator yang menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Contoh 2: Contoh ini menyinkronkan IAM MFA perangkat yang terkait dengan IAM pengguna **`Theresa`** dengan perangkat fisik yang memiliki nomor seri **`ABCD12345678`** dan yang menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Untuk API detailnya, lihat ResyncMfaDevicedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`SetDefaultPolicyVersion`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`SetDefaultPolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Kembalikan versi kebijakan

CLI

AWS CLI

Untuk menetapkan versi tertentu dari kebijakan yang ditentukan sebagai versi default kebijakan.

Contoh ini menetapkan `v2` versi kebijakan ARN yang `arn:aws:iam::123456789012:policy/MyPolicy` sebagai versi aktif default.

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat SetDefaultPolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menetapkan **v2** versi kebijakan ARN yang **arn:aws:iam::123456789012:policy/MyPolicy** sebagai versi aktif default.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Untuk API detailnya, lihat SetDefaultPolicyVersiondi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **TagRole** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`TagRole`.

CLI

AWS CLI

Untuk menambahkan tag ke peran

`tag-role`Perintah berikut menambahkan tag dengan nama Departemen ke peran yang ditentukan.

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai IAM sumber daya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat TagRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menambahkan tag ke Peran dalam Layanan Manajemen Identitas

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk API detailnya, lihat TagRoledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`TagUser`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`TagUser`.

CLI

AWS CLI

Untuk menambahkan tag ke pengguna

`tag-user`Perintah berikut menambahkan tag dengan Departemen terkait untuk pengguna tertentu.

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai IAM sumber daya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat TagUserdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menambahkan tag ke Pengguna di Layanan Manajemen Identitas

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk API detailnya, lihat TagUserdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UntagRole`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UntagRole`.

CLI

AWS CLI

Untuk menghapus tag dari peran

`untag-role`Perintah berikut menghapus tag apa pun dengan nama kunci 'Departemen' dari peran yang ditentukan.

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai IAM sumber daya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UntagRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus tag dari peran bernama "MyRoleName" dengan kunci tag sebagai “abac”. Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Untuk API detailnya, lihat UntagRoledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UntagUser`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UntagUser`.

CLI

AWS CLI

Untuk menghapus tag dari pengguna

`untag-user`Perintah berikut menghapus tag apa pun dengan nama kunci 'Departemen' dari pengguna yang ditentukan.

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai IAM sumber daya di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UntagUserdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus tag dari pengguna bernama “joe” dengan kunci tag sebagai “abac” dan “xyzw”. Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Untuk API detailnya, lihat UntagUserdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateAccessKey`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateAccessKey`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kunci akses

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  Aws::IAM::Model::StatusType status,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat UpdateAccessKeydi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mengaktifkan atau menonaktifkan kunci akses untuk pengguna IAM

`update-access-key`Perintah berikut menonaktifkan kunci akses yang ditentukan (ID kunci akses dan kunci akses rahasia) untuk IAM pengguna bernama. Bob

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Menonaktifkan kunci berarti tidak dapat digunakan untuk akses terprogram ke. AWS Namun, kuncinya masih tersedia dan dapat diaktifkan kembali.

Untuk informasi selengkapnya, lihat Mengelola kunci akses untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateAccessKeydi Referensi AWS CLI Perintah.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateAccessKey {

    private static StatusType statusType;

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessId> <status>\s

                Where:
```

```
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }

    public static void updateKey(IamClient iam, String username, String accessId,
 String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
                    .build();

            iam.updateAccessKey(request);
```

```
            System.out.printf("Successfully updated the status of access key %s
 to" +
                    "status %s for user %s", accessId, status, username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat UpdateAccessKeydi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Perbarui kunci akses.

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
```

```
    Status: StatusType.Inactive,
    UserName: userName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat UpdateAccessKeydi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat UpdateAccessKeydi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengubah status kunci akses **AKIAIOSFODNN7EXAMPLE** untuk IAM pengguna bernama **BobInactive**.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Untuk API detailnya, lihat UpdateAccessKeydi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
```

```
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
key_id
        )
        raise
```

- Untuk API detailnya, lihat UpdateAccessKey AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateAccountPasswordPolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateAccountPasswordPolicy`.

CLI

AWS CLI

Untuk mengatur atau mengubah kebijakan kata sandi akun saat ini

`update-account-password-policy`Perintah berikut menetapkan kebijakan kata sandi untuk memerlukan panjang minimum delapan karakter dan memerlukan satu atau lebih angka dalam kata sandi.

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

Perintah ini tidak menghasilkan output.

Perubahan pada kebijakan kata sandi akun memengaruhi kata sandi baru apa pun yang dibuat untuk IAM pengguna di akun. Perubahan kebijakan kata sandi tidak memengaruhi kata sandi yang ada.

Untuk informasi selengkapnya, lihat Menyetel kebijakan kata sandi akun untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateAccountPasswordPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui kebijakan kata sandi untuk akun dengan pengaturan yang ditentukan. Perhatikan bahwa parameter apa pun yang tidak termasuk dalam perintah tidak dibiarkan tidak dimodifikasi. Sebaliknya, mereka diatur ulang ke nilai default.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Untuk API detailnya, lihat UpdateAccountPasswordPolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateAssumeRolePolicy`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateAssumeRolePolicy`.

CLI

AWS CLI

Untuk memperbarui kebijakan kepercayaan untuk suatu IAM peran

`update-assume-role-policy`Perintah berikut memperbarui kebijakan kepercayaan untuk peran bernama`Test-Role`.

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

Perintah ini tidak menghasilkan output.

Kebijakan kepercayaan didefinisikan sebagai JSON dokumen dalam file test-role-trust-policy.json. (Nama file dan ekstensi tidak memiliki signifikansi.) Kebijakan kepercayaan harus menentukan kepala sekolah.

Untuk memperbarui kebijakan izin untuk peran, gunakan `put-role-policy` perintah.

Untuk informasi selengkapnya, lihat Membuat IAM peran di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateAssumeRolePolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui IAM peran yang diberi nama **`ClientRole`** dengan kebijakan kepercayaan baru, yang isinya berasal dari file**`ClientRolePolicy.json`**. Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** sakelar untuk berhasil memproses isi JSON file.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Untuk API detailnya, lihat UpdateAssumeRolePolicydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateGroup`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateGroup`.

CLI

AWS CLI

Untuk mengganti nama grup IAM

`update-group`Perintah berikut mengubah nama IAM grup `Test` menjadi`Test-1`.

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengganti nama grup IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengganti nama IAM grup **Testers** menjadi. **AppTesters**

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Contoh 2: Contoh ini mengubah jalur IAM grup **AppTesters** menjadi**/Org1/Org2/**. Ini mengubah ARN grup untuk**arn:aws:iam::123456789012:group/Org1/Org2/AppTesters**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Untuk API detailnya, lihat UpdateGroupdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **UpdateLoginProfile** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateLoginProfile`.

CLI

AWS CLI

Untuk memperbarui kata sandi untuk IAM pengguna

`update-login-profile`Perintah berikut membuat kata sandi baru untuk IAM pengguna bernama`Bob`.

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

Perintah ini tidak menghasilkan output.

Untuk menetapkan kebijakan kata sandi untuk akun, gunakan `update-account-password-policy` perintah. Jika kata sandi baru melanggar kebijakan kata sandi akun, perintah mengembalikan `PasswordPolicyViolation` kesalahan.

Jika kebijakan kata sandi akun memungkinkan mereka, IAM pengguna dapat mengubah kata sandi mereka sendiri menggunakan `change-password` perintah.

Simpan kata sandi di tempat yang aman. Jika kata sandi hilang, itu tidak dapat dipulihkan, dan Anda harus membuat yang baru menggunakan `create-login-profile` perintah.

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menetapkan kata sandi sementara baru untuk IAM pengguna**Bob**, dan mengharuskan pengguna untuk mengubah kata sandi saat pengguna masuk berikutnya.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Untuk API detailnya, lihat UpdateLoginProfiledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateOpenIdConnectProviderThumbprint`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateOpenIdConnectProviderThumbprint`.

CLI

AWS CLI

Untuk mengganti daftar sidik jari sertifikat server yang ada dengan daftar baru

Contoh ini memperbarui daftar cap jempol sertifikat untuk OIDC penyedia yang menggunakan ARN `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com` sidik jari baru.

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di AWS IAMPanduan Pengguna.

- Untuk API detailnya, lihat UpdateOpenIdConnectProviderThumbprintdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui daftar cap jempol sertifikat untuk OIDC penyedia yang menggunakan ARN **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`** sidik jari baru. OIDCPenyedia membagikan nilai baru ketika sertifikat yang terkait dengan penyedia berubah.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Untuk API detailnya, lihat UpdateOpenIdConnectProviderThumbprintdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateRole`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateRole`.

CLI

AWS CLI

Untuk mengubah deskripsi IAM peran atau durasi sesi

`update-role`Perintah berikut mengubah deskripsi IAM peran `production-role` menjadi `Main production role` dan menetapkan durasi sesi maksimum menjadi 12 jam.

```
aws iam update-role \
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui deskripsi peran dan nilai durasi sesi maksimum (dalam detik) yang sesi peran dapat diminta.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Untuk API detailnya, lihat UpdateRoledi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateRoleDescription`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateRoleDescription`.

CLI

AWS CLI

Untuk mengubah deskripsi IAM peran

`update-role`Perintah berikut mengubah deskripsi IAM peran `production-role` menjadi`Main production role`.

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

Output:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {}
                }
            ]
        },
```

```
        "Description": "Main production role"
    }
}
```

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateRoleDescriptiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui deskripsi IAM peran di akun Anda.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Untuk API detailnya, lihat UpdateRoleDescriptiondi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateSamlProvider`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateSamlProvider`.

CLI

AWS CLI

Untuk memperbarui dokumen metadata untuk penyedia yang ada SAML

Contoh ini memperbarui SAML penyedia IAM yang memiliki ARN dokumen SAML metadata baru dari file. `arn:aws:iam::123456789012:saml-provider/SAMLADFS SAMLMetaData.xml`

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Output:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia IAM SAML identitas di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateSamlProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui SAML penyedia IAM ARN yang **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`** memiliki dokumen SAML metadata baru dari file. **`SAMLMetaData.xml`** Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** sakelar untuk berhasil memproses isi JSON file.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Untuk API detailnya, lihat UpdateSamlProviderdi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateServerCertificate`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateServerCertificate`.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
                                           const Aws::String &newCertificateName,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);

    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }
```

```
    return result;
}
```

- Untuk API detailnya, lihat UpdateServerCertificatedi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mengubah jalur atau nama sertifikat server di AWS akun Anda

`update-server-certificate`Perintah berikut mengubah nama sertifikat dari `myServerCertificate` menjadi`myUpdatedServerCertificate`. Itu juga mengubah jalur ke `/cloudfront/` sehingga dapat diakses oleh CloudFront layanan Amazon. Perintah ini tidak menghasilkan output. Anda dapat melihat hasil pembaruan dengan menjalankan `list-server-certificates` perintah.

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengelola sertifikat server IAM di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Perbarui sertifikat server.

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat UpdateServerCertificatedi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};
```

```
iam.updateServerCertificate(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat UpdateServerCertificatedi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengganti nama sertifikat yang dinamai **MyServerCertificate** menjadi. **MyRenamedServerCertificate**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Contoh 2: Contoh ini memindahkan sertifikat yang diberi nama **MyServerCertificate** ke path /Org1/Org 2/. Ini mengubah sumber daya ARN untuk**arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Untuk API detailnya, lihat UpdateServerCertificatedi AWS Tools for PowerShell Referensi Cmdlet.

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
```

```
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Untuk API detailnya, lihat UpdateServerCertificatedi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateSigningCertificate`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateSigningCertificate`.

CLI

AWS CLI

Untuk mengaktifkan atau menonaktifkan sertifikat penandatanganan untuk pengguna IAM

`update-signing-certificate`Perintah berikut menonaktifkan sertifikat penandatanganan yang ditentukan untuk IAM pengguna bernama. Bob

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Untuk mendapatkan ID untuk sertifikat penandatanganan, gunakan `list-signing-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat penandatanganan di Panduan EC2 Pengguna Amazon.

- Untuk API detailnya, lihat UpdateSigningCertificatedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui sertifikat yang terkait dengan nama IAM pengguna **Bob** dan yang ID sertifikatnya si **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** untuk menandainya sebagai tidak aktif.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Untuk API detailnya, lihat UpdateSigningCertificatedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UpdateUser`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateUser`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
                             const Aws::String &newUserName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
```

```
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat UpdateUserdi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mengubah nama IAM pengguna

`update-user`Perintah berikut mengubah nama IAM pengguna Bob menjadi`Robert`.

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengganti nama grup IAM pengguna di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat UpdateUserdi Referensi AWS CLI Perintah.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String curName = args[0];
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
```

```
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat UpdateUserdi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Perbarui pengguna.

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
```

```
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk API detailnya, lihat UpdateUserdi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for JavaScript.
- Untuk API detailnya, lihat UpdateUserdi AWS SDK for JavaScript APIReferensi.

Kotlin

SDKuntuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun updateIAMUser(
    curName: String?,
    newName: String?,
) {
    val request =
        UpdateUserRequest {
            userName = curName
            newUserName = newName
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
    }
}
```

- Untuk API detailnya, lihat UpdateUser AWSSDKAPIreferensi Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengganti nama IAM pengguna **Bob** menjadi. **Robert**

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Contoh 2: Contoh ini mengubah jalur IAM Pengguna **Bob** ke**/Org1/Org2/**, yang secara efektif mengubah ARN untuk penggunaan**arn:aws:iam::123456789012:user/Org1/Org2/bob**.

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Untuk API detailnya, lihat UpdateUserdi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- Untuk API detailnya, lihat UpdateUser AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
# Updates an IAM user's name
#
# @param current_name [String] The current name of the user
# @param new_name [String] The new name of the user
def update_user_name(current_name, new_name)
  @iam_client.update_user(user_name: current_name, new_user_name: new_name)
  true
rescue StandardError => e
  @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
  false
end
```

- Untuk API detailnya, lihat UpdateUserdi AWS SDK for Ruby APIReferensi.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UploadServerCertificate`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UploadServerCertificate`.

CLI

AWS CLI

Untuk mengunggah sertifikat server ke AWS akun Anda

upload-server-certificatePerintah berikut mengunggah sertifikat server ke AWS akun Anda. Dalam contoh ini, sertifikat ada di file`public_key_cert_file.pem`, kunci pribadi terkait ada di file`my_private_key.pem`, dan rantai sertifikat yang disediakan oleh otoritas sertifikat (CA) ada di `my_certificate_chain_file.pem` file. Ketika file telah selesai diunggah, itu tersedia di bawah nama myServerCertificate. Parameter yang dimulai dengan `file://` memberi tahu perintah untuk membaca isi file dan menggunakannya sebagai nilai parameter alih-alih nama file itu sendiri.

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

Output:

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat Membuat, Mengunggah, dan Menghapus Sertifikat Server di panduan Menggunakan IAM.

- Untuk API detailnya, lihat UploadServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "node:path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
```

```
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

  return client.send(command);
};
```

- Untuk API detailnya, lihat UploadServerCertificatedi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengunggah sertifikat server baru ke IAM akun. File yang berisi badan sertifikat, kunci pribadi, dan (opsional) rantai sertifikat semuanya harus PEM dikodekan. Perhatikan bahwa parameter memerlukan konten sebenarnya dari file daripada nama file. Anda harus menggunakan parameter **`-Raw`** sakelar untuk berhasil memproses konten file.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk API detailnya, lihat UploadServerCertificatedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`UploadSigningCertificate`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UploadSigningCertificate`.

CLI

AWS CLI

Untuk mengunggah sertifikat penandatanganan untuk IAM pengguna

`upload-signing-certificate`Perintah berikut mengunggah sertifikat penandatanganan untuk IAM pengguna bernama`Bob`.

```
aws iam upload-signing-certificate \
    --user-name Bob \
    --certificate-body file://certificate.pem
```

Output:

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

Sertifikat dalam file bernama certificate.pem dalam format. PEM

Untuk informasi selengkapnya, lihat Membuat dan Mengunggah Sertifikat Penandatanganan Pengguna di IAM panduan Menggunakan.

- Untuk API detailnya, lihat UploadSigningCertificatedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengunggah sertifikat penandatanganan X.509 baru dan mengaitkannya dengan nama pengguna. IAM **Bob** File yang berisi badan sertifikat PEM dikodekan. **CertificateBody**Parameter memerlukan konten sebenarnya dari file sertifikat daripada nama file. Anda harus menggunakan parameter **-Raw** sakelar untuk berhasil memproses file.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
```

```
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Untuk API detailnya, lihat UploadSigningCertificatedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Skenario untuk IAM menggunakan AWS SDKs

Contoh kode berikut menunjukkan kepada Anda bagaimana menerapkan skenario umum IAM dengan AWS SDKs. Skenario ini menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi di dalam IAM atau dikombinasikan dengan yang lain Layanan AWS. Setiap skenario menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode.

Skenario menargetkan tingkat pengalaman menengah untuk membantu Anda memahami tindakan layanan dalam konteks.

Contoh

- Membangun dan mengelola layanan tangguh menggunakan AWS SDK
- Buat IAM grup dan tambahkan pengguna ke grup menggunakan AWS SDK
- Buat pengguna read-only dan read-write menggunakan IAM AWS SDK
- Mengelola kunci IAM akses menggunakan AWS SDK
- Mengelola IAM kebijakan menggunakan AWS SDK
- Mengelola IAM peran menggunakan AWS SDK
- Mengelola IAM akun Anda menggunakan AWS SDK
- Kembalikan versi IAM kebijakan menggunakan AWS SDK
- Bekerja dengan Pembuat IAM Kebijakan API menggunakan AWS SDK

Membangun dan mengelola layanan tangguh menggunakan AWS SDK

Contoh kode berikut menunjukkan cara membuat layanan web dengan beban seimbang yang mengembalikan rekomendasi buku, film, dan lagu. Contoh ini menunjukkan cara layanan tersebut merespons kegagalan, serta cara merestrukturisasi layanan agar lebih tangguh ketika terjadi kegagalan.

- Gunakan grup EC2 Auto Scaling Amazon untuk membuat instans Amazon Elastic Compute Cloud (AmazonEC2) berdasarkan template peluncuran dan untuk menyimpan jumlah instans dalam rentang yang ditentukan.
- Menangani dan mendistribusikan HTTP permintaan dengan Elastic Load Balancing.
- Memantau kondisi instans dalam grup Auto Scaling dan meneruskan permintaan hanya ke instans yang sehat.
- Jalankan server web Python pada setiap EC2 instance untuk menangani HTTP permintaan. Server web merespons dengan memberikan rekomendasi dan melakukan pemeriksaan kondisi.
- Menyimulasikan layanan yang direkomendasikan dengan tabel Amazon DynamoDB.
- Kontrol respons server web terhadap permintaan dan pemeriksaan kesehatan dengan memperbarui AWS Systems Manager parameter.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
static async Task Main(string[] args)
{
    _configuration = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("settings.json") // Load settings from .json file.
        .AddJsonFile("settings.local.json",
            true) // Optionally, load local settings.
```

```
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));
```

```
            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
    }

    /// <summary>
```

```
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
```

```
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
```

```
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            + "defines how the load balancer connects to instances. The load
balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
_autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
_elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
protocol, port, defaultVpc.VpcId);

        await
_elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
subnetIds, targetGroup);
        await
_autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
_elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
that the port is open...");
```

```
            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
 your default VPC must\n"
                    + "allows access from this computer. You can either add it
 automatically from this\n"
                    + "example or add it yourself using the AWS Management
 Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound traffic from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
 ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound SSH traffic for debugging from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
 ipString);
                }
```

```
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();
```

```
        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
```

```
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");
```

```
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
_elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
```

```
            await
_elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
_autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
_autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
_autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
_recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Buat kelas yang membungkus Auto Scaling dan tindakan AmazonEC2.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;
    private readonly ILogger<AutoScalerWrapper> _logger;

    private readonly string _instanceType = "";
```

```
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration,
        ILogger<AutoScalerWrapper> logger)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;
        _logger = logger;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];
```

```
        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
                                         "}," +
                                    "\"Action\": \"sts:AssumeRole\"" +
```

```
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
```

```
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
```

```
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
```

```
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
    /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        try
        {
            await CreateKeyPair(_keyPairName);
            await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName,
                _instanceProfileName, instancePolicyPath);

            var startServerText = await File.ReadAllTextAsync(startupScriptPath);
            var plainTextBytes =
System.Text.Encoding.UTF8.GetBytes(startServerText);

            var amiLatest = await _amazonSsm.GetParameterAsync(
                new GetParameterRequest() { Name = _amiParam });
            var amiId = amiLatest.Parameter.Value;
            var launchTemplateResponse = await
_amazonEc2.CreateLaunchTemplateAsync(
```

```
                new CreateLaunchTemplateRequest()
                {
                    LaunchTemplateName = _launchTemplateName,
                    LaunchTemplateData = new RequestLaunchTemplateData()
                    {
                        InstanceType = _instanceType,
                        ImageId = amiId,
                        IamInstanceProfile =
                            new

LaunchTemplateIamInstanceProfileSpecificationRequest()
                            {
                                Name = _instanceProfileName
                            },
                        KeyName = _keyPairName,
                        UserData = System.Convert.ToBase64String(plainTextBytes)
                    }
                });
            return launchTemplateResponse.LaunchTemplate;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.AlreadyExistsException")
            {
                _logger.LogError($"Could not create the template, the name
{_launchTemplateName} already exists. " +
                                 $"Please try again with a unique name.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while creating the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
    /// </summary>
```

```
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        try
        {
            var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
                new DescribeAvailabilityZonesRequest());
            return zoneResponse.AvailabilityZones.Select(z =>
 z.ZoneName).ToList();
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            _logger.LogError($"An Amazon EC2 error occurred while listing
 availability zones.: {ec2Exception.Message}");
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while listing availability
 zones.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
```

```
                        new
Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        try
        {
            var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
                new DescribeVpcsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("is-default", new List<string>() { "true" })
                    }
                });
            return vpcResponse.Vpcs[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "UnauthorizedOperation")
            {
                _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
            }
```

```
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the vpcs.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        try
        {
            var subnets = new List<Subnet>();
            var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
                new DescribeSubnetsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("vpc-id", new List<string>() { vpcId }),
                        new("availability-zone", availabilityZones),
                        new("default-for-az", new List<string>() { "true" })
                    }
                });

            // Get the entire list using the paginator.
            await foreach (var subnet in subnetPaginator.Subnets)
            {
                subnets.Add(subnet);
            }

            return subnets;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
```

```
            if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
            {
                _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the
subnets.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.NotFoundException")
            {
                _logger.LogError(
                    $"Could not delete the template, the name
{_launchTemplateName} was not found.");
            }

            throw;
        }
        catch (Exception ex)
```

```
        {
            _logger.LogError($"An error occurred while deleting the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
```

```
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        try
        {
```

```
            var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
                new DescribeIamInstanceProfileAssociationsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("instance-id", new List<string>() { instanceId })
                    },
                });
            return response.IamInstanceProfileAssociations[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while creating the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
```

```
    {
        try
        {
            await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
            // Allow time before resetting.
            Thread.Sleep(25000);

            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(25000);
            var instanceReady = false;
            var retries = 5;
            while (retries-- > 0 && !instanceReady)
            {
                var instancesPaginator =
                    _amazonSsm.Paginators.DescribeInstanceInformation(
                        new DescribeInstanceInformationRequest());
                // Get the entire list using the paginator.
                await foreach (var instance in
instancesPaginator.InstanceInformationList)
                {
                    instanceReady = instance.InstanceId == instanceId;
                    if (instanceReady)
                    {
                        break;
                    }
                }
            }
            Console.WriteLine("Waiting for instance to be running.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);
            Console.WriteLine("Instance ready.");
            Console.WriteLine($"Sending restart command to instance
{instanceId}");
            await _amazonSsm.SendCommandAsync(
                new SendCommandRequest()
                {
```

```
                    InstanceIds = new List<string>() { instanceId },
                    DocumentName = "AWS-RunShellScript",
                    Parameters = new Dictionary<string, List<string>>()
                    {
                        {
                            "commands",
                            new List<string>() { "cd / && sudo python3 server.py
80" }
                        }
                    }
                });
            Console.WriteLine($"Restarted the web server on instance
{instanceId}");
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
```

```
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
```

```
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }
```

```
    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
```

```
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
```

```
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
 InstanceStateName stateName)
```

```
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is in the specified state.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEc2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }
}
```

Membuat kelas yang menggabungkan tindakan Penyeimbangan Beban Elastis.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
```

```
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }

        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
```

```
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
```

```
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
    /// Create an Elastic Load Balancing load balancer that uses the specified
 subnets
    /// and forwards requests to the specified target group.
    /// </summary>
    /// <param name="name">The name for the new load balancer.</param>
    /// <param name="subnetIds">Subnets for the load balancer.</param>
    /// <param name="targetGroup">Target group for forwarded requests.</param>
    /// <returns>The new LoadBalancer object.</returns>
    public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
 List<string> subnetIds, TargetGroup targetGroup)
```

```
    {
        var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
            new CreateLoadBalancerRequest()
            {
                Name = name,
                Subnets = subnetIds
            });
        var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

        // Wait for load balancer to be available.
        var loadBalancerReady = false;
        while (!loadBalancerReady)
        {
            try
            {
                var describeResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                        new DescribeLoadBalancersRequest()
                        {
                            Names = new List<string>() { name }
                        });

                var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

                loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
            }
            catch (LoadBalancerNotFoundException)
            {
                loadBalancerReady = false;
            }
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
```

```
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
```

```
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }

    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
```

```
        {
            try
            {
                var groupResponse =
                    await
 _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
 targetArn });
                Console.WriteLine($"Deleted load balancing target group
 {groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

Membuat kelas yang menggunakan DynamoDB untuk menyimulasikan layanan yang direkomendasikan.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
```

```
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
 configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
```

```
                    }
                },
            KeySchema = new List<KeySchemaElement>()
                {
                    new KeySchemaElement()
                    {
                        AttributeName = "MediaType",
                        KeyType = KeyType.HASH
                    },
                    new KeySchemaElement()
                    {
                        AttributeName = "ItemId",
                        KeyType = KeyType.RANGE
                    }
                },
            ProvisionedThroughput = new ProvisionedThroughput()
            {
                ReadCapacityUnits = 5,
                WriteCapacityUnits = 5
            }
        };
        await _amazonDynamoDb.CreateTableAsync(createRequest);

        // Wait until the table is ACTIVE and then report success.
        Console.Write("\nWaiting for table to become active...");

        var request = new DescribeTableRequest
        {
            TableName = tableName
        };

        TableStatus status;
        do
        {
            Thread.Sleep(2000);

            var describeTableResponse = await
_amazonDynamoDb.DescribeTableAsync(request);
            status = describeTableResponse.Table.TableStatus;

            Console.Write(".");
        }
        while (status != "ACTIVE");
```

```
            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
```

```
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Membuat kelas yang mengabungkan tindakan Systems Manager.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";

    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
```

```
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for .NET APIReferensi.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer

- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
public class Main {
```

```
    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException,
 InterruptedException {
        Scanner in = new Scanner(System.in);
        Database database = new Database();
        AutoScaler autoScaler = new AutoScaler();
        LoadBalancer loadBalancer = new LoadBalancer();

        System.out.println(DASHES);
        System.out.println("Welcome to the demonstration of How to Build and
 Manage a Resilient Service!");
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("A - SETUP THE RESOURCES");
        System.out.println("Press Enter when you're ready to start deploying
resources.");
        in.nextLine();
        deploy(loadBalancer);
        System.out.println(DASHES);
        System.out.println(DASHES);
        System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        demo(loadBalancer);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("C - DELETE THE RESOURCES");
        System.out.println("""
                This concludes the demo of how to build and manage a resilient
service.
                To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
                that were created for this demo.
                """);

        System.out.println("\n Do you want to delete the resources (y/n)? ");
        String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

        if (userInput.equals("y")) {
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
```

```
   }

   // Deletes the AWS resources used in this example.
   private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
           throws IOException, InterruptedException {
       loadBalancer.deleteLoadBalancer(lbName);
       System.out.println("*** Wait 30 secs for resource to be deleted");
       TimeUnit.SECONDS.sleep(30);
       loadBalancer.deleteTargetGroup(targetGroupName);
       autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
       autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
       autoScaler.deleteTemplate(templateName);
       database.deleteTable(tableName);
   }

   private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
       Scanner in = new Scanner(System.in);
       System.out.println(
               """
                       For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                       to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                       against various kinds of failures.

                       Some of the resources create by this demo are:
                       \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                       \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                       \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                       \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                       """);

       System.out.println("Press Enter when you're ready.");
       in.nextLine();
       System.out.println(DASHES);

       System.out.println(DASHES);
```

```
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
```

```
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
```

```
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
```

```
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
```

```
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        System.out.println(
                """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
                       While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                       """);
        paramHelper.put(paramHelper.failureResponse, "static");

        System.out.println("""
                Now, sending a GET request to the load balancer endpoint returns
a static response.
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);
```

```
        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
```

```
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
```

```
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
```

```
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
target.target().id(),
                                    target.target().port(),
target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
```

```
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Buat kelas yang membungkus Auto Scaling dan tindakan AmazonEC2.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;

    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
```

```
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }

    /**
     * Replaces the profile associated with a running instance. After the profile
is
```

```
     * replaced, the instance is rebooted to ensure that it uses the new profile.
     * When
     * the instance is ready, Systems Manager is used to restart the Python web
     * server.
     */
    public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
            throws InterruptedException {
        // Create an IAM instance profile specification.
        software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
                .builder()
                .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                              // name.
                .build();

        // Replace the IAM instance profile association for the EC2 instance.
        ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
```

```
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
```

```
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }
```

```
   /*
    * Verify the default security group of the specified VPC allows ingress from
    * this
    * computer. This can be done by allowing ingress from this computer's IP
    * address. In some situations, such as connecting from a corporate network,
you
    * must instead specify a prefix list ID. You can also temporarily open the
port
    * to
    * any IP address while running this example. If you do, be sure to remove
    * public
    * access when you're done.
    *
    */
   public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
       boolean portIsOpen = false;
       GroupInfo groupInfo = new GroupInfo();
       try {
           Filter filter = Filter.builder()
                   .name("group-name")
                   .values("default")
                   .build();

           Filter filter1 = Filter.builder()
                   .name("vpc-id")
                   .values(VPC)
                   .build();

           DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                   .filters(filter, filter1)
                   .build();

           DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                   .describeSecurityGroups(securityGroupsRequest);
           String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
           groupInfo.setGroupName(securityGroup);

           for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
```

```
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
```

```
    * Scaling group.
    * The target group specifies how the load balancer forward requests to the
    * instances
    * in the group.
    */
   public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
       try {
           AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                   .autoScalingGroupName(asGroupName)
                   .targetGroupARNs(targetGroupARN)
                   .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
           System.out.println("Attached load balancer to " + asGroupName);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   // Creates an EC2 Auto Scaling group with the specified size.
   public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

       // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());
```

```
        String availabilityZones = String.join(",", availabilityZoneNames);
        LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
                .launchTemplateName(templateName)
                .version("$Default")
                .build();

        String[] zones = availabilityZones.split(",");
        CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
                .launchTemplate(specification)
                .availabilityZones(zones)
                .maxSize(groupSize)
                .minSize(groupSize)
                .autoScalingGroupName(autoScalingGroupName)
                .build();

        try {
            getAutoScalingClient().createAutoScalingGroup(groupRequest);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
```

```
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());
```

```
        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
        ListPoliciesRequest listPoliciesRequest =
ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
the policy.

software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
listEntitiesRequest =
software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
```

```
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
```

```
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Membuat kelas yang menggabungkan tindakan Penyeimbangan Beban Elastis.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
 targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
 DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();
```

```
        DescribeTargetGroupsResponse tgResponse =
getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
```

```
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
```

```
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
```

```
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");

            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
```

```
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .defaultActions(action)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

Membuat kelas yang menggunakan DynamoDB untuk menyimulasikan layanan yang direkomendasikan.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
```

```
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
e.getMessage());
        }
        return false;
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
     * hash key named 'MediaType' that defines the type of media recommended,
such
     * as
     * Book or Movie, and a range key named 'ItemId' that, combined with the
     * MediaType,
     * forms a unique identifier for the recommended item.
     */
    public void createTable(String tableName, String fileName) throws IOException
{
        // First check to see if the table exists.
        boolean doesExist = doesTableExist(tableName);
        if (!doesExist) {
            DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
            CreateTableRequest createTableRequest = CreateTableRequest.builder()
                    .tableName(tableName)
                    .attributeDefinitions(
                            AttributeDefinition.builder()
```

```
                                .attributeName("MediaType")
                                .attributeType(ScalarAttributeType.S)
                                .build(),
                        AttributeDefinition.builder()
                                .attributeName("ItemId")
                                .attributeType(ScalarAttributeType.N)
                                .build())
                .keySchema(
                        KeySchemaElement.builder()
                                .attributeName("MediaType")
                                .keyType(KeyType.HASH)
                                .build(),
                        KeySchemaElement.builder()
                                .attributeName("ItemId")
                                .keyType(KeyType.RANGE)
                                .build())
                .provisionedThroughput(
                        ProvisionedThroughput.builder()
                                .readCapacityUnits(5L)
                                .writeCapacityUnits(5L)
                                .build())
                .build();

        getDynamoDbClient().createTable(createTableRequest);
        System.out.println("Creating table " + tableName + "...");

        // Wait until the Amazon DynamoDB table is created.
        DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                .tableName(tableName)
                .build();

        WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Table " + tableName + " created.");

        // Add records to the table.
        populateTable(fileName, tableName);
    }
}

public void deleteTable(String tableName) {
    getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
    System.out.println("Table " + tableName + " deleted.");
```

```
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
 IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
 enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Membuat kelas yang mengabungkan tindakan Systems Manager.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
```

```
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Java 2.x APIReferensi.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate

- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

import {
  Scenario,
  parseScenarioArgs,
```

```
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "node:url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

Menyusun langkah-langkah untuk men-deploy semua sumber daya.

```
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";
```

```
import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { saveState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
```

```
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
```

```
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
```

```
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
```

```
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
```

```
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
```

```
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
```

```
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
```

```
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
```

```
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
```

```
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
```

```
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      }
      return MESSAGES.noIpRules;
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      }
      return MESSAGES.noIpRules;
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
```

```
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    }
    return false;
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
  saveState,
];
```

Menyusun langkah-langkah untuk menjalankan demo.

```
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
```

```
import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );

  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  state.targetHealthDescriptions = TargetHealthDescriptions;
```

```
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
```

```
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
```

```
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
```

```
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
```

```
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
```

```
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
```

```
        ShouldDecrementDesiredCapacity: false,
      }),
    );
  },
),
new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
healthCheckLoop,
loadBalancerLoop,
new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
  type: "confirm",
}),
new ScenarioAction("failOpenExit", (state) => {
  if (!state.failOpenConfirmation) {
    process.exit();
  }
}),
new ScenarioAction("failOpen", () => {
  const client = new SSMClient({});
  return client.send(
    new PutParameterCommand({
      Name: NAMES.ssmTableNameKey,
      Value: `fake-table-${Date.now()}`,
      Overwrite: true,
      Type: "String",
    }),
  );
}),
new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
healthCheckLoop,
loadBalancerLoop,
new ScenarioInput(
  "resetTableConfirmation",
  MESSAGES.demoResetTableConfirmation,
  { type: "confirm" },
),
new ScenarioAction("resetTableExit", (state) => {
  if (!state.resetTableConfirmation) {
    process.exit();
  }
}),
new ScenarioAction("resetTable", async () => {
  const client = new SSMClient({});
  await client.send(
    new PutParameterCommand({
```

```
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
```

```
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Menyusun langkah-langkah untuk menghancurkan semua sumber daya.

```
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
  RevokeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
```

```
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { loadState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  loadState,
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
```

```
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    }
    return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
    return MESSAGES.deletedKeyPair.replace(
      "${KEY_PAIR_NAME}",
      NAMES.keyPairName,
    );
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
```

```
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.detachedPolicyFromRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
```

```
        NAMES.instancePolicyName,
      );
    }
    return MESSAGES.deletedPolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.removedRoleFromInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
```

```
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
    return MESSAGES.deletedInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    );
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
    return MESSAGES.deletedInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
```

```
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
    return MESSAGES.deletedAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
    return MESSAGES.deletedLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    );
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
```

```
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
```

```
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    }
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
    return MESSAGES.detachedSsmOnlyRoleFromProfile
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
```

```
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
    return MESSAGES.detachedSsmOnlyCustomRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
```

```
    return MESSAGES.detachedSsmOnlyAWSRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
```

```
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
    return MESSAGES.deletedSsmOnlyRole.replace(
      "${ROLE_NAME}",
      NAMES.ssmOnlyRoleName,
    );
  }),
  new ScenarioAction(
    "revokeSecurityGroupIngress",
    async (
      /** @type {{ myIp: string, defaultSecurityGroup: { GroupId: string } }} */
state,
    ) => {
      const ec2Client = new EC2Client({});
```

```
      try {
        await ec2Client.send(
          new RevokeSecurityGroupIngressCommand({
            GroupId: state.defaultSecurityGroup.GroupId,
            CidrIp: `${state.myIp}/32`,
            FromPort: 80,
            ToPort: 80,
            IpProtocol: "tcp",
          }),
        );
      } catch (e) {
        state.revokeSecurityGroupIngressError = e;
      }
    },
  ),
  new ScenarioOutput("revokeSecurityGroupIngressResult", (state) => {
    if (state.revokeSecurityGroupIngressError) {
      console.error(state.revokeSecurityGroupIngressError);
      return MESSAGES.revokeSecurityGroupIngressError.replace(
        "${IP}",
        state.myIp,
      );
    }
    return MESSAGES.revokedSecurityGroupIngress.replace("${IP}", state.myIp);
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
 */
```

```
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    }
    console.log(err.name);
    throw err;
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
```

```
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for JavaScript APIReferensi.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs

- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
class Runner:
    """
    Manages the deployment, demonstration, and destruction of resources for the
 resilient service.
    """

    def __init__(
        self,
        resource_path: str,
        recommendation: RecommendationService,
        autoscaler: AutoScalingWrapper,
        loadbalancer: ElasticLoadBalancerWrapper,
        param_helper: ParameterHelper,
    ):
        """
        Initializes the Runner class with the necessary parameters.

        :param resource_path: The path to resource files used by this example,
 such as IAM policies and instance scripts.
        :param recommendation: An instance of the RecommendationService class.
        :param autoscaler: An instance of the AutoScaler class.
        :param loadbalancer: An instance of the LoadBalancer class.
        :param param_helper: An instance of the ParameterHelper class.
```

```
        """
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

        prefix = "doc-example-resilience"
        self.target_group_name = f"{prefix}-tg"
        self.load_balancer_name = f"{prefix}-lb"

    def deploy(self) -> None:
        """
        Deploys the resources required for the resilient service, including the
DynamoDB table,
        EC2 instances, Auto Scaling group, and load balancer.
        """
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        logging.info("Starting deployment of resources for the resilient
service.")

        logging.info(
            "Creating and populating DynamoDB table '%s'.",
            self.recommendation.table_name,
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)

        logging.info(
            "Creating an EC2 launch template with the startup script '%s'.",
            startup_script,
        )
        self.autoscaler.create_template(startup_script, instance_policy)

        logging.info(
            "Creating an EC2 Auto Scaling group across multiple Availability
Zones."
        )
```

```
        zones = self.autoscaler.create_autoscaling_group(3)

        logging.info("Creating variables that control the flow of the demo.")
        self.param_helper.reset()

        logging.info("Creating Elastic Load Balancing target group and load
 balancer.")

        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.target_group_name, self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            self.load_balancer_name, [subnet["SubnetId"] for subnet in subnets]
        )
        self.loadbalancer.create_listener(self.load_balancer_name, target_group)

        self.autoscaler.attach_load_balancer_target_group(target_group)

        logging.info("Verifying access to the load balancer endpoint.")
        endpoint = self.loadbalancer.get_endpoint(self.load_balancer_name)
        lb_success = self.loadbalancer.verify_load_balancer_endpoint(endpoint)
        current_ip_address = requests.get("http://
checkip.amazonaws.com").text.strip()

        if not lb_success:
            logging.warning(
                "Couldn't connect to the load balancer. Verifying that the port
 is open..."
            )
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                logging.warning(
                    "The default security group for your VPC must allow access
 from this computer."
                )
                if q.ask(
```

```
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success =
self.loadbalancer.verify_load_balancer_endpoint(endpoint)

        if lb_success:
            logging.info(
                "Load balancer is ready. Access it at: http://%s",
current_ip_address
            )
        else:
            logging.error(
                "Couldn't get a successful response from the load balancer
endpoint. Please verify your VPC and security group settings."
            )

    def demo_choices(self) -> None:
        """
        Presents choices for interacting with the deployed service, such as
sending requests to
        the load balancer or checking the health of the targets.
        """
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
```

```
        ]
        choice = 0
        while choice != 2:
            logging.info("Choose an action to interact with the service.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                logging.info("Sending a GET request to the load balancer
endpoint.")
                endpoint =
self.loadbalancer.get_endpoint(self.load_balancer_name)
                logging.info("GET http://%s", endpoint)
                response = requests.get(f"http://{endpoint}")
                logging.info("Response: %s", response.status_code)
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                logging.info("Checking the health of load balancer targets.")
                health =
self.loadbalancer.check_target_health(self.target_group_name)
                for target in health:
                    state = target["TargetHealth"]["State"]
                    logging.info(
                        "Target %s on port %d is %s",
                        target["Target"]["Id"],
                        target["Target"]["Port"],
                        state,
                    )
                    if state != "healthy":
                        logging.warning(
                            "%s: %s",
                            target["TargetHealth"]["Reason"],
                            target["TargetHealth"]["Description"],
                        )
                logging.info(
                    "Note that it can take a minute or two for the health check
to update."
                )
            elif choice == 2:
                logging.info("Proceeding to the next part of the demo.")

    def demo(self) -> None:
        """
        Runs the demonstration, showing how the service responds to different
failure scenarios
```

```
        and how a resilient architecture can keep the service running.
        """
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        logging.info("Resetting parameters to starting values for the demo.")
        self.param_helper.reset()

        logging.info(
            "Starting demonstration of the service's resilience under various
failure conditions."
        )
        self.demo_choices()

        logging.info(
            "Simulating failure by changing the Systems Manager parameter to a
non-existent table."
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info("Sending GET requests will now return failure codes.")
        self.demo_choices()

        logging.info("Switching to static response mode to mitigate failure.")
        self.param_helper.put(self.param_helper.failure_response, "static")
        logging.info("Sending GET requests will now return static responses.")
        self.demo_choices()

        logging.info("Restoring normal operation of the recommendation service.")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)

        logging.info(
            "Introducing a failure by assigning bad credentials to one of the
instances."
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
```

```
        logging.info(
            "Replacing instance profile with bad credentials for instance %s.",
            bad_instance_id,
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        logging.info(
            "Sending GET requests may return either a valid recommendation or a
static response."
        )
        self.demo_choices()

        logging.info("Implementing deep health checks to detect unhealthy
instances.")
        self.param_helper.put(self.param_helper.health_check, "deep")
        logging.info("Checking the health of the load balancer targets.")
        self.demo_choices()

        logging.info(
            "Terminating the unhealthy instance to let the auto scaler replace
it."
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        logging.info("The service remains resilient during instance
replacement.")
        self.demo_choices()

        logging.info("Simulating a complete failure of the recommendation
service.")
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info(
            "All instances will report as unhealthy, but the service will still
return static responses."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self, automation=False) -> None:
        """
        Destroys all resources created for the demo, including the load balancer,
Auto Scaling group,
```

```
        EC2 instances, and DynamoDB table.
        """
        logging.info(
            "This concludes the demo. Preparing to clean up all AWS resources
 created during the demo."
        )
        if automation:
            cleanup = True
        else:
            cleanup = q.ask(
                "Do you want to clean up all demo resources? (y/n) ", q.is_yesno
            )

        if cleanup:
            logging.info("Deleting load balancer and related resources.")
            self.loadbalancer.delete_load_balancer(self.load_balancer_name)
            self.loadbalancer.delete_target_group(self.target_group_name)
            self.autoscaler.delete_autoscaling_group(self.autoscaler.group_name)
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            logging.info("Deleting DynamoDB table and other resources.")
            self.recommendation.destroy()
        else:
            logging.warning(
                "Resources have not been deleted. Ensure you clean them up
 manually to avoid unexpected charges."
            )


def main() -> None:
    """
    Main function to parse arguments and run the appropriate actions for the
 demo.
    """
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
```

```
        help="The action to take for the demo. When 'all' is specified, resources
are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../workflows/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    logging.info("Starting the Resilient Service demo.")

    prefix = "doc-example-resilience"

    # Service Clients
    ddb_client = boto3.client("dynamodb")
    elb_client = boto3.client("elbv2")
    autoscaling_client = boto3.client("autoscaling")
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")

    # Wrapper instantiations
    recommendation = RecommendationService(
        "doc-example-recommendation-service", ddb_client
    )
    autoscaling_wrapper = AutoScalingWrapper(
        prefix,
        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    )
    elb_wrapper = ElasticLoadBalancerWrapper(elb_client)
    param_helper = ParameterHelper(recommendation.table_name, ssm_client)

    # Demo invocation
    runner = Runner(
        args.resource_path,
```

```
        recommendation,
        autoscaling_wrapper,
        elb_wrapper,
        param_helper,
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    logging.info("Demo completed successfully.")


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Buat kelas yang membungkus Auto Scaling dan tindakan AmazonEC2.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.
```

```
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_policy(self, policy_file: str, policy_name: str) -> str:
        """
        Creates a new IAM policy or retrieves the ARN of an existing policy.

        :param policy_file: The path to a JSON file that contains the policy
definition.
        :param policy_name: The name to give the created policy.
        :return: The ARN of the created or existing policy.
        """
        with open(policy_file) as file:
```

```
            policy_doc = file.read()

        try:
            response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=policy_doc
            )
            policy_arn = response["Policy"]["Arn"]
            log.info(f"Policy '{policy_name}' created successfully. ARN:
 {policy_arn}")
            return policy_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the policy already exists, get its ARN
                response = self.iam_client.get_policy(
                    PolicyArn=f"arn:aws:iam::{self.account_id}:policy/
{policy_name}"
                )
                policy_arn = response["Policy"]["Arn"]
                log.info(f"Policy '{policy_name}' already exists. ARN:
 {policy_arn}")
                return policy_arn
            log.error(f"Full error:\n\t{err}")

    def create_role(self, role_name: str, assume_role_doc: dict) -> str:
        """
        Creates a new IAM role or retrieves the ARN of an existing role.

        :param role_name: The name to give the created role.
        :param assume_role_doc: The assume role policy document that specifies
 which
                                entities can assume the role.
        :return: The ARN of the created or existing role.
        """
        try:
            response = self.iam_client.create_role(
                RoleName=role_name,
 AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            role_arn = response["Role"]["Arn"]
            log.info(f"Role '{role_name}' created successfully. ARN: {role_arn}")
            return role_arn

        except ClientError as err:
```

```
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the role already exists, get its ARN
                response = self.iam_client.get_role(RoleName=role_name)
                role_arn = response["Role"]["Arn"]
                log.info(f"Role '{role_name}' already exists. ARN: {role_arn}")
                return role_arn
            log.error(f"Full error:\n\t{err}")

    def attach_policy(
        self,
        role_name: str,
        policy_arn: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> None:
        """
        Attaches an IAM policy to a role and optionally attaches additional AWS-
managed policies.

        :param role_name: The name of the role to attach the policy to.
        :param policy_arn: The ARN of the policy to attach.
        :param aws_managed_policies: A tuple of AWS-managed policy names to
 attach to the role.
        """
        try:
            self.iam_client.attach_role_policy(RoleName=role_name,
 PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info(f"Attached policy {policy_arn} to role {role_name}.")
        except ClientError as err:
            log.error(f"Failed to attach policy {policy_arn} to role
 {role_name}.")
            log.error(f"Full error:\n\t{err}")

    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
```

```
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
```

```
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn


    def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
            if not response["IamInstanceProfileAssociations"]:
                log.info(f"No instance profile found for instance
{instance_id}.")
            profile_data = response["IamInstanceProfileAssociations"][0]
            log.info(f"Retrieved instance profile for instance {instance_id}.")
            return profile_data
        except ClientError as err:
            log.error(
                f"Failed to retrieve instance profile for instance
{instance_id}."
            )
```

```
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidInstanceID.NotFound":
                log.error(f"The instance ID '{instance_id}' does not exist.")
            log.error(f"Full error:\n\t{err}")


    def replace_instance_profile(
        self,
        instance_id: str,
        new_instance_profile_name: str,
        profile_association_id: str,
    ) -> None:
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to restart.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)

            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info("Rebooting instance %s.", instance_id)
            waiter = self.ec2_client.get_waiter("instance_running")
            log.info("Waiting for instance %s to be running.", instance_id)
```

```
            waiter.wait(InstanceIds=[instance_id])
            log.info("Instance %s is now running.", instance_id)

            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info(f"Restarted the Python web server on instance
'{instance_id}'.")
        except ClientError as err:
            log.error("Failed to replace instance profile.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidAssociationID.NotFound":
                log.error(
                    f"Association ID '{profile_association_id}' does not exist."
                    "Please check the association ID and try again."
                )
            if error_code == "InvalidInstanceId":
                log.error(
                    f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
                    f"Please verify the instance ID and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
```

```
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )


    def create_key_pair(self, key_pair_name: str) -> None:
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to create key pair {key_pair_name}.")
            if error_code == "InvalidKeyPair.Duplicate":
                log.error(f"A key pair with the name '{key_pair_name}' already
exists.")
            log.error(f"Full error:\n\t{err}")
```

```
    def delete_key_pair(self) -> None:
        """
        Deletes a key pair.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            log.error(f"Couldn't delete key pair '{self.key_pair_name}'.")
            log.error(f"Full error:\n\t{err}")
        except FileNotFoundError as err:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
            log.error(f"Full error:\n\t{err}")


    def create_template(
        self, server_startup_script_file: str, instance_policy_file: str
    ) -> Dict[str, Any]:
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            # Create key pair and instance profile
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
```

```
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )

            # Read the startup script
            with open(server_startup_script_file) as file:
                start_server_script = file.read()

            # Get the latest AMI ID
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]

            # Create the launch template
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
            )
        except ClientError as err:
            log.error(f"Failed to create launch template
{self.launch_template_name}.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
                log.info(
                    f"Launch template {self.launch_template_name} already exists,
nothing to do."
                )
            log.error(f"Full error:\n\t{err}")
        return template
```

```
    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            log.error(f"Full error:\n\t{err}")


    def get_availability_zones(self) -> List[str]:
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
            log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
        except ClientError as err:
            log.error("Failed to retrieve availability zones.")
            log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def create_autoscaling_group(self, group_size: int) -> List[str]:
        """
```

```
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
                f"Created EC2 Auto Scaling group {self.group_name} with
availability zones {zones}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            if error_code == "AlreadyExists":
                log.info(
                    f"EC2 Auto Scaling group {self.group_name} already exists,
nothing to do."
                )
            else:
                log.error(f"Failed to create EC2 Auto Scaling group
{self.group_name}.")
                log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def get_instances(self) -> List[str]:
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: A list of instance IDs in the Auto Scaling group.
        """
```

```
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
            log.info(
                f"Retrieved {len(instance_ids)} instances for Auto Scaling group
{self.group_name}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to retrieve instances for Auto Scaling group
{self.group_name}."
            )
            if error_code == "ResourceNotFound":
                log.error(f"The Auto Scaling group '{self.group_name}' does not
exist.")
            log.error(f"Full error:\n\t{err}")
        else:
            return instance_ids


    def terminate_instance(self, instance_id: str, decrementsetting=False) ->
None:
        """
        Terminates an instance in an EC2 Auto Scaling group. After an instance is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        :param decrementsetting: If True, do not replace terminated instances.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id,
                ShouldDecrementDesiredCapacity=decrementsetting,
            )
            log.info("Terminated instance %s.", instance_id)

            # Adding a waiter to ensure the instance is terminated
            waiter = self.ec2_client.get_waiter("instance_terminated")
```

```
            log.info("Waiting for instance %s to be terminated...", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info(
                f"Instance '{instance_id}' has been terminated and will be
replaced."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to terminate instance '{instance_id}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to terminate the instance again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            log.error(f"Full error:\n\t{err}")

    def attach_load_balancer_target_group(
        self, lb_target_group: Dict[str, Any]
    ) -> None:
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forwards requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
```

```
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
            )
            if error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            elif error_code == "ServiceLinkedRoleFailure":
                log.error(
                    "The operation failed because the service-linked role is not
ready or does not exist. "
                    "Check that the service-linked role exists and is correctly
configured."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_autoscaling_group(self, group_name: str) -> None:
        """
        Terminates all instances in the group, then deletes the EC2 Auto Scaling
group.

        :param group_name: The name of the group to delete.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
```

```
                    self.terminate_instance(inst_id)

                # Wait for all instances to be terminated
                if instance_ids:
                    waiter = self.ec2_client.get_waiter("instance_terminated")
                    log.info("Waiting for all instances to be terminated...")
                    waiter.wait(InstanceIds=instance_ids)
                    log.info("All instances have been terminated.")
            else:
                log.info(f"No groups found named '{group_name}'! Nothing to do.")
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to delete the group again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_default_vpc(self) -> Dict[str, Any]:
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error("Failed to retrieve the default VPC.")
            if error_code == "UnauthorizedOperation":
                log.error(
```

```
                    "You do not have the necessary permissions to describe VPCs.
 "
                    "Ensure that your AWS IAM user or role has the correct
 permissions."
                )
            elif error_code == "InvalidParameterValue":
                log.error(
                    "One or more parameters are invalid. Check the request
 parameters."
                )

            log.error(f"Full error:\n\t{err}")
        else:
            if "Vpcs" in response and response["Vpcs"]:
                log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
                return response["Vpcs"][0]
            else:
                pass


    def verify_inbound_port(
        self, vpc: Dict[str, Any], port: int, ip_address: str
    ) -> Tuple[Dict[str, Any], bool]:
        """
        Verify the default security group of the specified VPC allows ingress
 from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
 you
        must instead specify a prefix list ID. You can also temporarily open the
 port to
        any IP address while running this example. If you do, be sure to remove
 public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
        :return: The default security group of the specified VPC, and a value
 that indicates
                 whether the specified port is open.
        """
        try:
```

```
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info(f"Found default security group {sec_group['GroupId']}.")

            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info(f"Found inbound rule: {ip_perm}")
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
                    if not port_is_open:
                        log.info(
                            f"The inbound rule does not appear to be open to
either this computer's IP "
                            f"address of {ip_address}, to all IP addresses
(0.0.0.0/0), or to a prefix list ID."
                        )
                    else:
                        break
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to verify inbound rule for port {port} for VPC
{vpc['VpcId']}."
            )
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    f"The specified VPC ID '{vpc['VpcId']}' does not exist.
Please check the VPC ID."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return sec_group, port_is_open
```

```
   def open_inbound_port(self, sec_group_id: str, port: int, ip_address: str) ->
None:
       """
       Add an ingress rule to the specified security group that allows access on
the
       specified port from the specified IP address.

       :param sec_group_id: The ID of the security group to modify.
       :param port: The port to open.
       :param ip_address: The IP address that is granted access.
       """
       try:
           self.ec2_client.authorize_security_group_ingress(
               GroupId=sec_group_id,
               CidrIp=f"{ip_address}/32",
               FromPort=port,
               ToPort=port,
               IpProtocol="tcp",
           )
           log.info(
               "Authorized ingress to %s on port %s from %s.",
               sec_group_id,
               port,
               ip_address,
           )
       except ClientError as err:
           error_code = err.response["Error"]["Code"]
           log.error(
               f"Failed to authorize ingress to security group '{sec_group_id}'
on port {port} from {ip_address}."
           )
           if error_code == "InvalidGroupId.Malformed":
               log.error(
                   "The security group ID is malformed. "
                   "Please verify that the security group ID is correct."
               )
           elif error_code == "InvalidPermission.Duplicate":
               log.error(
                   "The specified rule already exists in the security group. "
                   "Check the existing rules for this security group."
               )
           log.error(f"Full error:\n\t{err}")
```

```
   def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
       """
       Gets the default subnets in a VPC for a specified list of Availability
Zones.

       :param vpc_id: The ID of the VPC to look up.
       :param zones: The list of Availability Zones to look up.
       :return: The list of subnets found.
       """
       # Ensure that 'zones' is a list, even if None is passed
       if zones is None:
           zones = []
       try:
           paginator = self.ec2_client.get_paginator("describe_subnets")
           page_iterator = paginator.paginate(
               Filters=[
                   {"Name": "vpc-id", "Values": [vpc_id]},
                   {"Name": "availability-zone", "Values": zones},
                   {"Name": "default-for-az", "Values": ["true"]},
               ]
           )

           subnets = []
           for page in page_iterator:
               subnets.extend(page["Subnets"])

           log.info("Found %s subnets for the specified zones.", len(subnets))
           return subnets
       except ClientError as err:
           log.error(
               f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
{zones}."
           )
           error_code = err.response["Error"]["Code"]
           if error_code == "InvalidVpcID.NotFound":
               log.error(
                   "The specified VPC ID does not exist. "
                   "Please check the VPC ID and try again."
               )
           # Add more error-specific handling as needed
           log.error(f"Full error:\n\t{err}")
```

Membuat kelas yang menggabungkan tindakan Penyeimbangan Beban Elastis.

```
class ElasticLoadBalancerWrapper:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, elb_client: boto3.client):
        """
        Initializes the LoadBalancer class with the necessary parameters.
        """
        self.elb_client = elb_client


    def create_target_group(
        self, target_group_name: str, protocol: str, port: int, vpc_id: str
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing target group. The target group
 specifies how
        the load balancer forwards requests to instances in the group and how
 instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
 times and
        lower thresholds. In production, you might want to decrease the
 sensitivity of
        your health checks to avoid unwanted failures.

        :param target_group_name: The name of the target group to create.
        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
```

```
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info(f"Created load balancing target group
'{target_group_name}'.")
            return target_group
        except ClientError as err:
            log.error(
                f"Couldn't create load balancing target group
'{target_group_name}'."
            )
            error_code = err.response["Error"]["Code"]

            if error_code == "DuplicateTargetGroupName":
                log.error(
                    f"Target group name {target_group_name} already exists. "
                    "Check if the target group already exists."
                    "Consider using a different name or deleting the existing
target group if appropriate."
                )
            elif error_code == "TooManyTargetGroups":
                log.error(
                    "Too many target groups exist in the account. "
                    "Consider deleting unused target groups to create space for
new ones."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_target_group(self, target_group_name) -> None:
        """
        Deletes the target group.
        """
        try:
            # Describe the target group to get its ARN
            response =
self.elb_client.describe_target_groups(Names=[target_group_name])
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]

            # Delete the target group
```

```
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
            log.info("Deleted load balancing target group %s.",
target_group_name)

            # Use a custom waiter to wait until the target group is no longer
available
            self.wait_for_target_group_deletion(self.elb_client, tg_arn)
            log.info("Target group %s successfully deleted.", target_group_name)

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete target group '{target_group_name}'.")
            if error_code == "TargetGroupNotFound":
                log.error(
                    "Load balancer target group either already deleted or never
existed. "
                    "Verify the name and check that the resource exists in the
AWS Console."
                )
            elif error_code == "ResourceInUseException":
                log.error(
                    "Target group still in use by another resource. "
                    "Ensure that the target group is no longer associated with
any load balancers or resources.",
                )
            log.error(f"Full error:\n\t{err}")

    def wait_for_target_group_deletion(
        self, elb_client, target_group_arn, max_attempts=10, delay=30
    ):
        for attempt in range(max_attempts):
            try:

elb_client.describe_target_groups(TargetGroupArns=[target_group_arn])
                print(
                    f"Attempt {attempt + 1}: Target group {target_group_arn}
still exists."
                )
            except ClientError as e:
                if e.response["Error"]["Code"] == "TargetGroupNotFound":
                    print(
                        f"Target group {target_group_arn} has been successfully
deleted."
                    )
```

```
                    return
                else:
                    raise
            time.sleep(delay)
        raise TimeoutError(
            f"Target group {target_group_arn} was not deleted after {max_attempts
* delay} seconds."
        )


    def create_load_balancer(
        self,
        load_balancer_name: str,
        subnet_ids: List[str],
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param load_balancer_name: The name of the load balancer to create.
        :param subnet_ids: A list of subnets to associate with the load balancer.
        :return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info(f"Created load balancer '{load_balancer_name}'.")

            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info(
                f"Waiting for load balancer '{load_balancer_name}' to be
available..."
            )
            waiter.wait(Names=[load_balancer_name])
            log.info(f"Load balancer '{load_balancer_name}' is now available!")

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to create load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
```

```
            )

            if error_code == "DuplicateLoadBalancerNameException":
                log.error(
                    f"A load balancer with the name '{load_balancer_name}'
already exists. "
                    "Load balancer names must be unique within the AWS region. "
                    "Please choose a different name and try again."
                )
            if error_code == "TooManyLoadBalancersException":
                log.error(
                    "The maximum number of load balancers has been reached in
this account and region. "
                    "You can delete unused load balancers or request an increase
in the service quota from AWS Support."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return load_balancer


    def create_listener(
        self,
        load_balancer_name: str,
        target_group: Dict[str, Any],
    ) -> Dict[str, Any]:
        """
        Creates a listener for the specified load balancer that forwards requests
to the
        specified target group.

        :param load_balancer_name: The name of the load balancer to create a
listener for.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created listener.
        """
        try:
            # Retrieve the load balancer ARN
            load_balancer_response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            load_balancer_arn = load_balancer_response["LoadBalancers"][0][
```

```
                "LoadBalancerArn"
            ]

            # Create the listener
            response = self.elb_client.create_listener(
                LoadBalancerArn=load_balancer_arn,
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                f"Created listener to forward traffic from load balancer
'{load_balancer_name}' to target group '{target_group['TargetGroupName']}'."
            )
            return response["Listeners"][0]
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to add a listener on '{load_balancer_name}' for target
group '{target_group['TargetGroupName']}'."
            )

            if error_code == "ListenerNotFoundException":
                log.error(
                    f"The listener could not be found for the load balancer
'{load_balancer_name}'. "
                    "Please check the load balancer name and target group
configuration."
                )
            if error_code == "InvalidConfigurationRequestException":
                log.error(
                    f"The configuration provided for the listener on load
balancer '{load_balancer_name}' is invalid. "
                    "Please review the provided protocol, port, and target group
settings."
                )
            log.error(f"Full error:\n\t{err}")
```

```
    def delete_load_balancer(self, load_balancer_name) -> None:
        """
        Deletes a load balancer.

        :param load_balancer_name: The name of the load balancer to delete.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[load_balancer_name])
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Couldn't delete load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    f"The load balancer '{load_balancer_name}' does not exist. "
                    "Please check the name and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def get_endpoint(self, load_balancer_name) -> str:
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            return response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            log.error(
```

```
                f"Couldn't get the endpoint for load balancer
{load_balancer_name}"
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Verify load balancer name and ensure it exists in the AWS
console."
                )
            log.error(f"Full error:\n\t{err}")

    @staticmethod
    def verify_load_balancer_endpoint(endpoint) -> bool:
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.

        :param endpoint: The endpoint to verify.
        :return: True if the GET request is successful, False otherwise.
        """
        retries = 3
        verified = False
        while not verified and retries > 0:
            try:
                lb_response = requests.get(f"http://{endpoint}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    verified = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
                )
                retries -= 1
                time.sleep(10)
        return verified

    def check_target_health(self, target_group_name: str) -> List[Dict[str,
Any]]:
```

```
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            log.error(f"Couldn't check health of {target_group_name} target(s).")
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Load balancer associated with the target group was not
found. "
                    "Ensure the load balancer exists, is in the correct AWS
region, and "
                    "that you have the necessary permissions to access it.",
                )
            elif error_code == "TargetGroupNotFoundException":
                log.error(
                    "Target group was not found. "
                    "Verify the target group name, check that it exists in the
correct region, "
                    "and ensure it has not been deleted or created in a different
account.",
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return health_response["TargetHealthDescriptions"]
```

Membuat kelas yang menggunakan DynamoDB untuk menyimulasikan layanan yang direkomendasikan.

```
class RecommendationService:
```

```
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
movies,
    and songs.
    """

    def __init__(self, table_name: str, dynamodb_client: boto3.client):
        """
        Initializes the RecommendationService class with the necessary
parameters.

        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    def create(self) -> Dict[str, Any]:
        """
        Creates a DynamoDB table to use as a recommendation service. The table
has a
        hash key named 'MediaType' that defines the type of media recommended,
such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        :raises RecommendationServiceError: If the table creation fails.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
```

```
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be done.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file: str) -> None:
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        :raises RecommendationServiceError: If the table population fails.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self) -> None:
        """
        Deletes the recommendations table.

        :raises RecommendationServiceError: If the table deletion fails.
        """
        try:
```

```
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Membuat kelas yang mengabungkan tindakan Systems Manager.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table: str = "doc-example-resilient-architecture-table"
    failure_response: str = "doc-example-resilient-architecture-failure-response"
    health_check: str = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name: str, ssm_client: boto3.client):
        """
        Initializes the ParameterHelper class with the necessary parameters.

        :param table_name: The name of the DynamoDB table that is used as a
 recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name
```

```
    def reset(self) -> None:
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name: str, value: str) -> None:
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        :raises ParameterHelperError: If the parameter value cannot be set.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting parameter %s to '%s'.", name, value)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to set parameter {name}.")
            if error_code == "ParameterLimitExceeded":
                log.error(
                    "The parameter limit has been exceeded. "
                    "Consider deleting unused parameters or request a limit
increase."
                )
            elif error_code == "ParameterAlreadyExists":
                log.error(
                    "The parameter already exists and overwrite is set to False.
"
                    "Use Overwrite=True to update the parameter."
                )
            log.error(f"Full error:\n\t{err}")
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Buat IAM grup dan tambahkan pengguna ke grup menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Buat grup dan berikan izin akses Amazon S3 penuh untuk itu.
- Buat pengguna baru tanpa izin untuk mengakses Amazon S3.
- Tambahkan pengguna ke grup dan tunjukkan bahwa mereka sekarang memiliki izin untuk Amazon S3, lalu bersihkan sumber daya.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;
```

```
    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
```

```
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
```

```
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
```

```
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
```

```
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
```

```
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
```

```
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
```

```
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
 ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
```

```
        var listRolePoliciesPaginator =
_IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
```

```
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
 policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
   /// <summary>
   /// Add or update an inline policy document that is embedded in an IAM user.
   /// </summary>
   /// <param name="userName">The name of the IAM user.</param>
   /// <param name="policyName">The name of the IAM policy.</param>
   /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
   {
       var request = new PutUserPolicyRequest
       {
           UserName = userName,
           PolicyName = policyName,
           PolicyDocument = policyDocument
       };

       var response = await _IAMService.PutUserPolicyAsync(request);
       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }

   /// <summary>
   /// Wait for a new access key to be ready to use.
   /// </summary>
   /// <param name="accessKeyId">The Id of the access key.</param>
   /// <returns>A boolean value indicating the success of the action.</returns>
   public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
   {
       var keyReady = false;

       do
       {
           try
           {
               var response = await _IAMService.GetAccessKeyLastUsedAsync(
                   new GetAccessKeyLastUsedRequest { AccessKeyId =
accessKeyId });
               if (response.UserName is not null)
               {
                   keyReady = true;
               }
           }
```

```
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}


using Microsoft.Extensions.Configuration;

namespace IAMGroups;

public class IAMGroups
{
    private static ILogger logger = null!;

    // Represents JSON code for AWS full access policy for Amazon Simple
    // Storage Service (Amazon S3).
    private const string S3FullAccessPolicyDocument = "{" +
        " \"Statement\" : [{" +
            " \"Action\" : [\"s3:*\"]," +
            " \"Effect\" : \"Allow\"," +
            " \"Resource\" : \"*\"" +
        "}]" +
    "}";


    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
```

```
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMGroups>();

        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var groupUserName = configuration["GroupUserName"];
        var groupName = configuration["GroupName"];
        var groupPolicyName = configuration["GroupPolicyName"];
        var groupBucketName = configuration["GroupBucketName"];

        var wrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayGroupsOverview();
        uiWrapper.PressEnter();

        // Create an IAM group.
        uiWrapper.DisplayTitle("Create IAM group");
        Console.WriteLine("Let's begin by creating a new IAM group.");
        var group = await wrapper.CreateGroupAsync(groupName);

        // Add an inline IAM policy to the group.
        uiWrapper.DisplayTitle("Add policy to group");
        Console.WriteLine("Add an inline policy to the group that allows members
to have full access to");
        Console.WriteLine("Amazon Simple Storage Service (Amazon S3) buckets.");

        await wrapper.PutGroupPolicyAsync(group.GroupName, groupPolicyName,
S3FullAccessPolicyDocument);

        uiWrapper.PressEnter();

        // Now create a new user.
        uiWrapper.DisplayTitle("Create an IAM user");
        Console.WriteLine("Now let's create a new IAM user.");
        var groupUser = await wrapper.CreateUserAsync(groupUserName);
```

```
        // Add the new user to the group.
        uiWrapper.DisplayTitle("Add the user to the group");
        Console.WriteLine("Adding the user to the group, which will give the user
the same permissions as the group.");
        await wrapper.AddUserToGroupAsync(groupUser.UserName, group.GroupName);

        Console.WriteLine($"User, {groupUser.UserName}, has been added to the
group, {group.GroupName}.");
        uiWrapper.PressEnter();

        Console.WriteLine("Now that we have created a user, and added the user to
the group, let's create an IAM access key.");

        // Create access and secret keys for the user.
        var accessKey = await wrapper.CreateAccessKeyAsync(groupUserName);
        Console.WriteLine("Key created.");
        uiWrapper.WaitABit(15, "Waiting for the access key to be ready for
use.");

        uiWrapper.DisplayTitle("List buckets");
        Console.WriteLine("To prove that the user has access to Amazon S3, list
the S3 buckets for the account.");

        var s3Client = new AmazonS3Client(accessKey.AccessKeyId,
accessKey.SecretAccessKey);
        var stsClient = new
AmazonSecurityTokenServiceClient(accessKey.AccessKeyId,
accessKey.SecretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client, stsClient);

        var buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
            });
        }

        // Show that the user also has write access to Amazon S3 by creating
```

```
        // a new bucket.
        uiWrapper.DisplayTitle("Create a bucket");
        Console.WriteLine("Since group members have full access to Amazon S3,
let's create a bucket.");
        var success = await s3Wrapper.PutBucketAsync(groupBucketName);

        if (success)
        {
            Console.WriteLine($"Successfully created the bucket:
{groupBucketName}.");
        }

        uiWrapper.PressEnter();

        Console.WriteLine("Let's list the user's S3 buckets again to show the new
bucket.");

        buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("First delete the bucket we created.");
        await s3Wrapper.DeleteBucketAsync(groupBucketName);

        Console.WriteLine($"Now remove the user, {groupUserName}, from the group,
{groupName}.");
        await wrapper.RemoveUserFromGroupAsync(groupUserName, groupName);

        Console.WriteLine("Delete the user's access key.");
        await wrapper.DeleteAccessKeyAsync(accessKey.AccessKeyId, groupUserName);

        // Now we can safely delete the user.
        Console.WriteLine("Now we can delete the user.");
        await wrapper.DeleteUserAsync(groupUserName);
```

```
        uiWrapper.PressEnter();

        Console.WriteLine("Now we will delete the IAM policy attached to the
 group.");
        await wrapper.DeleteGroupPolicyAsync(groupName, groupPolicyName);

        Console.WriteLine("Now we delete the IAM group.");
        await wrapper.DeleteGroupAsync(groupName);

        uiWrapper.PressEnter();

        Console.WriteLine("The IAM groups demo has completed.");

        uiWrapper.PressEnter();
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
```

```
    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
```

```
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;
```

```
public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
 s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
 to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
 role.");
```

```
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
```

```
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for .NET APIReferensi.
  - AddUserToGroup
  - AttachRolePolicy
  - CreateAccessKey
  - CreateGroup
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeleteGroup
  - DeleteGroupPolicy
  - DeleteUser
  - PutGroupPolicy
  - RemoveUserFromGroup

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Buat pengguna read-only dan read-write menggunakan IAM AWS SDK

Contoh kode berikut menunjukkan cara membuat pengguna dan melampirkan kebijakan kepada mereka.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan IAM pengguna untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat dua IAM pengguna.
- Lampirkan kebijakan bagi satu pengguna untuk mendapatkan dan meletakkan objek di bucket Amazon S3.
- Lampirkan kebijakan bagi pengguna kedua untuk mendapatkan objek dari bucket.
- Dapatkan izin berbeda ke bucket berdasarkan kredensyal pengguna.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM pengguna.

```
import logging
import time

import boto3
from botocore.exceptions import ClientError

import access_key_wrapper
import policy_wrapper
```

```
logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user



def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user



def list_users():
    """
    Lists the users in the current account.
```

```
    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise
```

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

Buat fungsi yang membungkus tindakan IAM kebijakan.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
```

```
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Buat fungsi yang membungkus tindakan kunci IAM akses.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
```

```
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Gunakan fungsi pembungkus untuk membuat pengguna dengan kebijakan berbeda dan menggunakan kredensialnya untuk mengakses bucket Amazon S3.

```
def usage_demo():
    """
    Shows how to manage users, keys, and policies.
    This demonstration creates two users: one user who can put and get objects in
 an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
 permitted
    to perform.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
    print(f"Created two IAM users: {user_read_writer.name} and
 {user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
```

```
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
    )
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
```

```
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )

    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
```

```
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
    detach_policy(user_reader.name, read_policy.arn)
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API

- AttachUserPolicy
- CreateAccessKey
- CreatePolicy
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteUser
- DetachUserPolicy
- ListUsers
- UpdateUser

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Mengelola kunci IAM akses menggunakan AWS SDK

Contoh kode berikut menunjukkan cara mengelola kunci akses.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan IAM pengguna untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat dan daftar kunci akses.
- Cari tahu kapan dan bagaimana kunci akses terakhir digunakan.
- Perbarui dan hapus kunci akses.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan kunci IAM akses.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.
```

```
    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Gunakan fungsi pembungkus untuk melakukan tindakan kunci akses untuk pengguna saat ini.

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
    if len(all_keys) == 2:
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
 {new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
 {last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
```

```
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - CreateAccessKey
  - DeleteAccessKey
  - GetAccessKeyLastUsed
  - ListAccessKeys
  - UpdateAccessKey

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Mengelola IAM kebijakan menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Buat dan daftar kebijakan.
- Buat dan dapatkan versi kebijakan.
- Kembalikan kebijakan ke versi sebelumnya.
- Hapus kebijakan.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM kebijakan.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
```

```
        raise
    else:
        return policy



def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
```

```
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version



def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.
```

```
    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
```

```
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Gunakan fungsi pembungkus untuk membuat kebijakan, memperbarui versi, dan mendapatkan informasi tentangnya.

```
def usage_demo():
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::made-up-bucket-name"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
```

```
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersion
  - GetPolicyVersion
  - ListPolicies
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Mengelola IAM peran menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Buat IAM peran.
- Lampirkan dan lepaskan kebijakan untuk suatu peran.
- Hapus peran.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM peran.

```
import json
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
```

```
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role



def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

Gunakan fungsi pembungkus untuk membuat peran, lalu lampirkan dan lepaskan kebijakan.

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
```

```
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
"batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
    pprint.pprint(role.assume_role_policy_document)
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - AttachRolePolicy
  - CreateRole
  - DeleteRole
  - DetachRolePolicy

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Mengelola IAM akun Anda menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Dapatkan dan perbarui alias akun.
- Hasilkan laporan pengguna dan kredensyal.
- Dapatkan ringkasan penggunaan akun.
- Dapatkan detail untuk semua pengguna, grup, peran, dan kebijakan di akun Anda, termasuk hubungan mereka satu sama lain.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan IAM akun.

```
import logging
import pprint
import sys
import time
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
```

```
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]



def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise



def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
```

```
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

Fungsi pembungkus panggilan untuk mengubah alias akun dan untuk mendapatkan laporan tentang akun.

```
def usage_demo():
```

```
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
```

```
    cred_lines = [
        line.split(",")[:col_count] for line in
 cred_report.decode("utf-8").split("\n")
    ]
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
 ")
    if see_pw_policy.lower() == "y":
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
```

```
            print("Password policy deleted.")

    print("The SAML providers for your account are:")
    list_saml_providers(10)

    print("-" * 88)
    print("Thanks for watching.")
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialReport
  - GetAccountAuthorizationDetails
  - GetAccountSummary
  - GetCredentialReport
  - ListAccountAliases

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Kembalikan versi IAM kebijakan menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Dapatkan daftar versi kebijakan secara berurutan berdasarkan tanggal.
- Temukan versi kebijakan default.
- Jadikan versi kebijakan sebelumnya sebagai default.
- Hapus versi default yang lama.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
```

```
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

- Untuk API detailnya, lihat topik berikut AWS SDKuntuk Referensi Python (Boto3). API
  - DeletePolicyVersion
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Bekerja dengan Pembuat IAM Kebijakan API menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Buat IAM kebijakan dengan menggunakan berorientasi objekAPI.
- Gunakan Pembuat IAM Kebijakan API dengan IAM layanan ini.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Contoh menggunakan impor berikut.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

Buat kebijakan berbasis waktu.

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
```

```
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.DATE_GREATER_THAN)

.key("aws:CurrentTime")

.value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.DATE_LESS_THAN)

.key("aws:CurrentTime")

.value("2020-06-30T23:59:59Z")))
                                .build();

                // Use an IamPolicyWriter to write out the JSON string to a more
readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
                                .build());
        }
```

Buat kebijakan dengan beberapa kondisi.

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

.addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

.addAction("dynamodb:BatchWriteItem")

.addResource("arn:aws:dynamodb:*:*:table/table-name")
```

```
 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                  List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))

 .key("dynamodb:Select")

 .value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Gunakan prinsipal dalam kebijakan.

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
                                                .addPrincipal(IamPrincipal.ALL)

 .addResource("arn:aws:s3:::BUCKETNAME/*")

 .addResource("arn:aws:s3:::BUCKETNAME")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.ARN_NOT_EQUALS)

 .key("aws:PrincipalArn")

 .value("arn:aws:iam::444455556666:user/user-name")))
```

```
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Izinkan akses lintas akun.

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)

 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                .key("s3:x-amz-
acl")
                                                                .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Membangun dan meng-upload file`IamPolicy`.

```
        public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
                // Build the policy.
                IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                                .addStatement(IamStatement.builder()
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
```

```
                                                                    + ":table/
exampleTableName")
                                                .build())
                                .build();
                // Upload the policy.
                iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
                return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

Unduh dan kerjakan dengan file`IamPolicy`.

```
        public String createNewBasedOnExistingPolicyExample(IamClient iam, String
 accountID, String policyName,
                        String newPolicyName) {

                String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
 policyName;
                GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
 r.policyArn(policyArn));

                String policyVersion =
 getPolicyResponse.policy().defaultVersionId();
                GetPolicyVersionResponse getPolicyVersionResponse = iam
                                .getPolicyVersion(r ->
 r.policyArn(policyArn).versionId(policyVersion));

                // Create an IamPolicy instance from the JSON string returned
 from IAM.
                String decodedPolicy =
 URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
                                StandardCharsets.UTF_8);
                IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

                /*
                 * All IamPolicy components are immutable, so use the copy method
 that creates a
                 * new instance that
                 * can be altered in the same method call.
                 *
```

```
                 * Add the ability to get an item from DynamoDB as an additional
action.
                 */
                IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

                // Create a new statement that replaces the original statement.
                IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

                // Upload the new policy. IAM now has both policies.
                iam.createPolicy(r -> r.policyName(newPolicyName)
                                .policyDocument(newPolicy.toJson()));

                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- Untuk informasi selengkapnya, lihat AWS SDK for Java 2.x Panduan Developer.
- Untuk API detailnya, lihat topik berikut di AWS SDK for Java 2.x APIReferensi.
  - CreatePolicy
  - GetPolicy
  - GetPolicyVersion

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Contoh kode untuk AWS STS menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan AWS STS kit pengembangan AWS perangkat lunak (SDK).

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain Layanan AWS.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Contoh kode

- Contoh dasar untuk AWS STS menggunakan AWS SDKs
  - Tindakan untuk AWS STS menggunakan AWS SDKs
    - Gunakan AssumeRole dengan AWS SDK atau CLI
    - Gunakan AssumeRoleWithWebIdentity dengan CLI
    - Gunakan DecodeAuthorizationMessage dengan CLI
    - Gunakan GetFederationToken dengan CLI
    - Gunakan GetSessionToken dengan AWS SDK atau CLI
- Skenario untuk AWS STS menggunakan AWS SDKs
  - Asumsikan IAM peran yang membutuhkan MFA token dengan AWS STS menggunakan AWS SDK
  - Membangun URL dengan AWS STS untuk pengguna federasi menggunakan AWS SDK
  - Dapatkan token sesi yang membutuhkan MFA token dengan AWS STS menggunakan AWS SDK

Contoh dasar untuk AWS STS menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar AWS Security Token Service dengan AWS SDKs.

Contoh

- Tindakan untuk AWS STS menggunakan AWS SDKs
  - Gunakan AssumeRole dengan AWS SDK atau CLI
  - Gunakan AssumeRoleWithWebIdentity dengan CLI
  - Gunakan DecodeAuthorizationMessage dengan CLI
  - Gunakan GetFederationToken dengan CLI

- Gunakan GetSessionToken dengan AWS SDK atau CLI

Tindakan untuk AWS STS menggunakan AWS SDKs

Contoh kode berikut menunjukkan bagaimana melakukan AWS STS tindakan individu dengan AWS SDKs. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Kutipan ini menyebut AWS STS API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Anda dapat melihat tindakan dalam konteks diSkenario untuk AWS STS menggunakan AWS SDKs .

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkap, lihat AWS Security Token Service APIReferensi.

Contoh

- Gunakan AssumeRole dengan AWS SDK atau CLI
- Gunakan AssumeRoleWithWebIdentity dengan CLI
- Gunakan DecodeAuthorizationMessage dengan CLI
- Gunakan GetFederationToken dengan CLI
- Gunakan GetSessionToken dengan AWS SDK atau CLI

Gunakan **`AssumeRole`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AssumeRole`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Asumsikan IAM peran yang membutuhkan MFA token
- Membangun a URL untuk pengguna federasi

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
```

```
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for .NET APIReferensi.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
```

```
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        ech o"Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk API detailnya, lihat AssumeRoledi Referensi AWS CLI Perintah.

C++

SDKuntuk C ++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for C++ APIReferensi.

CLI

AWS CLI

Untuk mengambil peran

`assume-role`Perintah berikut mengambil satu set kredensi jangka pendek untuk peran tersebut. IAM `s3-access-example`

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Output:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
```

```
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

Output dari perintah berisi kunci akses, kunci rahasia, dan token sesi yang dapat Anda gunakan untuk AWS mengautentikasi.

Untuk AWS CLI digunakan, Anda dapat mengatur profil bernama yang terkait dengan peran. Ketika Anda menggunakan profil, AWS CLI akan memanggil assume-role dan mengelola kredensi untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan IAM peran AWS CLI dalam Panduan AWS CLI Pengguna.

- Untuk API detailnya, lihat AssumeRoledi Referensi AWS CLI Perintah.

Java

SDKuntuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
```

```
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
```

```
                    roleSessionName - An identifier for the assumed role session
(for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
exTime);
```

```
        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for Java 2.x APIReferensi.

JavaScript

SDKuntuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat klien.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Asumsikan IAM peran.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
```

```
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for JavaScript APIReferensi.

SDKuntuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });
```

```
//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for JavaScript APIReferensi.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan satu set kredensi sementara (kunci akses, kunci rahasia, dan token sesi) yang dapat digunakan selama satu jam untuk mengakses AWS sumber daya yang biasanya tidak dapat diakses oleh pengguna yang meminta. Kredensil yang dikembalikan memiliki izin yang diizinkan oleh kebijakan akses dari peran yang diasumsikan dan kebijakan yang diberikan (Anda tidak dapat menggunakan kebijakan yang disediakan untuk memberikan izin melebihi yang ditentukan oleh kebijakan akses peran yang diasumsikan).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Contoh 2: Mengembalikan satu set kredensi sementara, berlaku selama satu jam, yang memiliki izin yang sama yang ditentukan dalam kebijakan akses peran yang diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Contoh 3: Mengembalikan satu set kredensi sementara yang memasok nomor seri dan token yang dihasilkan dari kredenal yang MFA terkait dengan pengguna yang digunakan untuk mengeksekusi cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Contoh 4: Mengembalikan satu set kredensi sementara yang telah mengambil peran yang ditentukan dalam akun pelanggan. Untuk setiap peran yang dapat diasumsikan oleh pihak ketiga, akun pelanggan harus membuat peran menggunakan pengidentifikasi yang harus diteruskan dalam ExternalId parameter - setiap kali peran diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Untuk API detailnya, lihat AssumeRoledi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Asumsikan IAM peran yang memerlukan MFA token dan gunakan kredensil sementara untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk API detailnya, lihat AssumeRole AWSSDKReferensi Python (Boto3). API

Ruby

SDKuntuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Untuk API detailnya, lihat AssumeRoledi AWS SDK for Ruby APIReferensi.

Rust

SDKuntuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Untuk API detailnya, lihat AssumeRole AWSSDKuntuk API referensi Rust.

Swift

SDKuntuk Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- Untuk API detailnya, lihat AssumeRole AWSSDKAPIreferensi Swift.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`AssumeRoleWithWebIdentity`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AssumeRoleWithWebIdentity`.

CLI

AWS CLI

Untuk mendapatkan kredensi jangka pendek untuk peran yang diautentikasi dengan Identitas Web (2."0) OAuth

`assume-role-with-web-identity`Perintah berikut mengambil satu set kredensi jangka pendek untuk peran tersebut. IAM `app1` Permintaan diautentikasi dengan menggunakan token identitas web yang disediakan oleh penyedia identitas web yang ditentukan. Dua kebijakan tambahan diterapkan pada sesi untuk lebih membatasi apa yang dapat dilakukan pengguna. Kredensi yang dikembalikan kedaluwarsa satu jam setelah dibuat.

```
aws sts assume-role-with-web-identity \
    --duration-seconds 3600 \
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/
webidentitydemopolicy2" \
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

Output:

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A"
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
```

```
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    }
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

Untuk informasi selengkapnya, lihat Meminta Kredensial Keamanan Sementara di Panduan Pengguna.AWS IAM

- Untuk API detailnya, lihat AssumeRoleWithWebIdentitydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan kumpulan kredensional sementara, berlaku selama satu jam, untuk pengguna yang telah diautentikasi dengan penyedia identitas Login with Amazon. Kredensi mengasumsikan kebijakan akses yang terkait dengan peran yang diidentifikasi oleh peran tersebut. ARN Secara opsional, Anda dapat meneruskan JSON kebijakan ke parameter -Policy yang selanjutnya menyempurnakan izin akses (Anda tidak dapat memberikan izin lebih banyak daripada yang tersedia di izin yang terkait dengan peran tersebut). Nilai yang diberikan ke - WebIdentityToken adalah pengidentifikasi pengguna unik yang dikembalikan oleh penyedia identitas.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Untuk API detailnya, lihat AssumeRoleWithWebIdentitydi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`DecodeAuthorizationMessage`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DecodeAuthorizationMessage`.

CLI

AWS CLI

Untuk memecahkan kode pesan otorisasi yang dikodekan dikembalikan sebagai tanggapan atas permintaan

`decode-authorization-message`Contoh berikut menerjemahkan informasi tambahan tentang status otorisasi permintaan dari pesan yang disandikan yang dikembalikan sebagai tanggapan atas permintaan Amazon Web Services.

```
aws sts decode-authorization-message \
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3
Ojau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKE
```

Output:

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
\"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
\":\"DENY\",\"principals\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE
\"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
\":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
\"}]},\"conditions\":{\"items\":[]}}]},\"failures\":{\"items\":[]},
\"context\":{\"principal\":{\"id\":\"AROA123456789EXAMPLE:Ana\",\"arn
\":\"arn:aws:sts::111122223333:assumed-role/Developer/Ana\"},\"action\":
\"RunInstances\",\"resource\":\"arn:aws:ec2:us-east-1:111122223333:instance/*
\",\"conditions\":{\"items\":[{\"key\":\"ec2:MetadataHttpPutResponseHopLimit\",
\"values\":{\"items\":[{\"value\":\"2\"}]}},{\"key\":\"ec2:InstanceMarketType
\",\"values\":{\"items\":[{\"value\":\"on-demand\"}]}},{\"key\":\"aws:Resource
```

```
\",\"values\":{\"items\":[{\"value\":\"instance/*\"}]}},{\"key\":\"aws:Account
\",\"values\":{\"items\":[{\"value\":\"111122223333\"}]}},{\"key\":
\"ec2:AvailabilityZone\",\"values\":{\"items\":[{\"value\":\"us-east-1f\"}]}},
{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items\":[{\"value\":\"false\"}]}},
{\"key\":\"ec2:IsLaunchTemplateResource\",\"values\":{\"items\":[{\"value\":
\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":{\"items\":[{\"value
\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values\":{\"items\":
[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values\":{\"items
\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":{\"items
\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items\":
[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items
\":[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items
\":[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":
[{\"value\":\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}}"
}
```

Untuk informasi selengkapnya, lihat Logika evaluasi kebijakan di Panduan AWS IAM Pengguna.

- Untuk API detailnya, lihat DecodeAuthorizationMessagedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mendekode informasi tambahan yang terkandung dalam konten pesan disandikan yang disediakan yang dikembalikan sebagai tanggapan atas permintaan. Informasi tambahan dikodekan karena rincian status otorisasi dapat merupakan informasi istimewa yang tidak boleh dilihat oleh pengguna yang meminta tindakan.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Untuk API detailnya, lihat DecodeAuthorizationMessagedi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetFederationToken`** dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetFederationToken`.

CLI

AWS CLI

Untuk mengembalikan satu set kredensi keamanan sementara menggunakan kredensi kunci akses IAM pengguna

`get-federation-token`Contoh berikut mengembalikan satu set kredensi keamanan sementara (terdiri dari ID kunci akses, kunci akses rahasia, dan token keamanan) untuk pengguna. Anda harus memanggil `GetFederationToken` operasi menggunakan kredensi keamanan jangka panjang dari pengguna. IAM

```
aws sts get-federation-token \
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

Isi dari `myfile.json`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

Output:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
    },
    "PackedPolicySize": 36
}
```

Untuk informasi selengkapnya, lihat Meminta Kredensial Keamanan Sementara di Panduan Pengguna.AWS IAM

- Untuk API detailnya, lihat GetFederationTokendi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Meminta token federasi yang valid selama satu jam menggunakan “Bob” sebagai nama pengguna federasi. Nama ini dapat digunakan untuk mereferensikan nama pengguna

federasi dalam kebijakan berbasis sumber daya (seperti kebijakan bucket Amazon S3). IAMKebijakan yang disediakan, dalam JSON format, digunakan untuk mencakupkan izin yang tersedia bagi IAM pengguna. Kebijakan yang diberikan tidak dapat memberikan izin lebih dari yang diberikan kepada pengguna yang meminta, dengan izin akhir untuk pengguna federasi menjadi set yang paling ketat berdasarkan persimpangan kebijakan yang disahkan dan kebijakan pengguna. IAM

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Untuk API detailnya, lihat GetFederationTokendi AWS Tools for PowerShell Referensi Cmdlet.

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Gunakan **`GetSessionToken`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetSessionToken`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Dapatkan token sesi yang membutuhkan MFA token

CLI

AWS CLI

Untuk mendapatkan satu set kredensi jangka pendek untuk identitas IAM

`get-session-token`Perintah berikut mengambil satu set kredensi jangka pendek untuk IAM identitas yang membuat panggilan. Kredensi yang dihasilkan dapat digunakan untuk permintaan di mana otentikasi multi-faktor (MFA) diperlukan oleh kebijakan. Kredensialnya kedaluwarsa 15 menit setelah dibuat.

```
aws sts get-session-token \
    --duration-seconds 900 \
```

```
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Output:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat Meminta Kredensial Keamanan Sementara di Panduan Pengguna.AWS IAM

- Untuk API detailnya, lihat GetSessionTokendi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan sebuah **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensil sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensil lainnya, gunakan parameter - ProfileName atau - AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Output:

```
AccessKeyId                          Expiration
 SecretAccessKey                     SessionToken
-----------                          ----------
 ---------------                     ------------
EXAMPLEACCESSKEYID                   2/16/2015 9:12:28 PM
 examplesecretaccesskey...           SamPleTokeN.....
```

Contoh 2: Mengembalikan sebuah **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensi sementara yang valid selama satu jam. Kredensi yang digunakan untuk membuat permintaan diperoleh dari profil yang ditentukan.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 3: Mengembalikan **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredenal sementara yang valid selama satu jam menggunakan nomor identifikasi MFA perangkat yang terkait dengan akun yang kredensialnya ditentukan dalam profil 'myprofilename' dan nilai yang disediakan oleh perangkat.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Untuk API detailnya, lihat GetSessionTokendi AWS Tools for PowerShell Referensi Cmdlet.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan token sesi dengan meneruskan MFA token dan gunakan untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
```

```
    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk API detailnya, lihat GetSessionToken AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Skenario untuk AWS STS menggunakan AWS SDKs

Contoh kode berikut menunjukkan kepada Anda bagaimana menerapkan skenario umum AWS STS dengan AWS SDKs. Skenario ini menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi di dalam AWS STS atau dikombinasikan dengan yang lain Layanan AWS. Setiap skenario menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode.

Skenario menargetkan tingkat pengalaman menengah untuk membantu Anda memahami tindakan layanan dalam konteks.

Contoh

- Asumsikan IAM peran yang membutuhkan MFA token dengan AWS STS menggunakan AWS SDK
- Membangun URL dengan AWS STS untuk pengguna federasi menggunakan AWS SDK
- Dapatkan token sesi yang membutuhkan MFA token dengan AWS STS menggunakan AWS SDK

Asumsikan IAM peran yang membutuhkan MFA token dengan AWS STS menggunakan AWS SDK

Contoh kode berikut menunjukkan bagaimana untuk mengambil peran yang membutuhkan MFA token.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan IAM pengguna untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat IAM peran yang memberikan izin untuk mencantumkan bucket Amazon S3.
- Buat IAM pengguna yang memiliki izin untuk mengambil peran hanya ketika MFA kredensil disediakan.
- Daftarkan MFA perangkat untuk pengguna.
- Asumsikan peran dan gunakan kredensil sementara untuk membuat daftar bucket S3.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna, daftarkan MFA perangkat, dan buat peran yang memberikan izin untuk membuat daftar bucket S3. Pengguna hanya memiliki hak untuk mengambil peran.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role and requires
 MFA.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.
```

```
    For demonstration purposes, the user is created in the same account as the
role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
```

```
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")

    user.create_policy(
        PolicyName=unique_name("user-policy"),
```

```
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

Tunjukkan bahwa mengasumsikan peran tanpa MFA token tidak diperbolehkan.

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
results
    in an AccessDenied error.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
    try:
```

```
        sts_client.assume_role(RoleArn=assume_role_arn,
 RoleSessionName=session_name)
        raise RuntimeError("Expected AccessDenied error.")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Got AccessDenied.")
        else:
            raise
```

Asumsikan peran yang memberikan izin untuk mencantumkan bucket S3, meneruskan MFA token yang diperlukan, dan tunjukkan bahwa bucket dapat dicantumkan.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
```

```
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Hancurkan sumber daya yang dibuat untuk demo.

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Jalankan skenario ini dengan menggunakan fungsi yang ditentukan sebelumnya.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- Untuk API detailnya, lihat AssumeRole AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Membangun URL dengan AWS STS untuk pengguna federasi menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Buat IAM peran yang memberikan akses hanya-baca ke sumber daya Amazon S3 akun saat ini.
- Dapatkan token keamanan dari titik akhir AWS federasi.
- Membangun sebuah URL yang dapat digunakan untuk mengakses konsol dengan kredenal federasi.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat peran yang memberikan akses hanya-baca ke sumber daya S3 akun saat ini.

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
                         that has the permission to create a role.
    :return: The newly created role.
    """
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
```

```
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return role
```

Dapatkan token keamanan dari titik akhir AWS federasi dan buat URL yang dapat digunakan untuk mengakses konsol dengan kredensi federasi.

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
 Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
 that
       can be used to assume a role with limited permissions.
    2. Uses the temporary credentials to request a sign-in token from the
       AWS federation endpoint.
    3. Builds a URL that can be used in a browser to navigate to the AWS
 federation
       endpoint, includes the sign-in token for authentication, and redirects to
       the AWS Management Console with permissions defined by the role that was
       specified in step 1.
```

```
   :param assume_role_arn: The role that specifies the permissions that are
granted.
                           The current user must have permission to assume the
role.
   :param session_name: The name for the STS session.
   :param issuer: The organization that issues the URL.
   :param sts_client: A Boto3 STS instance that can assume the role.
   :return: The federated URL.
   """
   response = sts_client.assume_role(
       RoleArn=assume_role_arn, RoleSessionName=session_name
   )
   temp_credentials = response["Credentials"]
   print(f"Assumed role {assume_role_arn} and got temporary credentials.")

   session_data = {
       "sessionId": temp_credentials["AccessKeyId"],
       "sessionKey": temp_credentials["SecretAccessKey"],
       "sessionToken": temp_credentials["SessionToken"],
   }
   aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

   # Make a request to the AWS federation endpoint to get a sign-in token.
   # The requests.get function URL-encodes the parameters and builds the query
string
   # before making the request.
   response = requests.get(
       aws_federated_signin_endpoint,
       params={
           "Action": "getSigninToken",
           "SessionDuration": str(datetime.timedelta(hours=12).seconds),
           "Session": json.dumps(session_data),
       },
   )
   signin_token = json.loads(response.text)
   print(f"Got a sign-in token from the AWS sign-in federation endpoint.")

   # Make a federated URL that can be used to sign into the AWS Management
Console.
   query_string = urllib.parse.urlencode(
       {
           "Action": "login",
           "Issuer": issuer,
```

```
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

Hancurkan sumber daya yang dibuat untuk demo.

```
def teardown(role):
    """
    Removes all resources created during setup.

    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

Jalankan skenario ini dengan menggunakan fungsi yang ditentukan sebelumnya.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
```

```
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- Untuk API detailnya, lihat AssumeRole AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Dapatkan token sesi yang membutuhkan MFA token dengan AWS STS menggunakan AWS SDK

Contoh kode berikut menunjukkan cara mendapatkan token sesi yang membutuhkan MFA token.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan IAM pengguna untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat IAM peran yang memberikan izin untuk mencantumkan bucket Amazon S3.
- Buat IAM pengguna yang memiliki izin untuk mengambil peran hanya ketika MFA kredensil disediakan.
- Daftarkan MFA perangkat untuk pengguna.

- Berikan MFA kredensil untuk mendapatkan token sesi dan gunakan kredensil sementara untuk membuat daftar bucket S3.

Python

SDKuntuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di Repositori Contoh Kode AWS.

Buat IAM pengguna, daftarkan MFA perangkat, dan buat peran yang memberikan izin agar pengguna mencantumkan bucket S3 hanya jika MFA kredensil digunakan.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")
```

```
    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
```

```
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

Dapatkan kredenal sesi sementara dengan meneruskan MFA token, dan gunakan kredensialnya untuk mencantumkan bucket S3 untuk akun tersebut.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]
```

```
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Hancurkan sumber daya yang dibuat untuk demo.

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Jalankan skenario ini dengan menggunakan fungsi yang ditentukan sebelumnya.

```
def usage_demo():
    """Drives the demonstration."""
```

```
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- Untuk API detailnya, lihat GetSessionToken AWSSDKReferensi Python (Boto3). API

Untuk daftar lengkap panduan AWS SDK pengembang dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang SDK versi sebelumnya.

Keamanan di IAM dan AWS STS

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan Program AWS Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku untuk AWS Identity and Access Management (IAM), lihat AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS Identity and Access Management (IAM) dan AWS Security Token Service (AWS STS). Topik berikut menunjukkan cara mengonfigurasi IAM dan AWS STS memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan IAM sumber daya Anda.

Konten

- AWS kebijakan terkelola untuk AWS Identity and Access Management Access Analyzer
- Fitur keamanan di luar IAM

AWS kredensi keamanan

Ketika Anda berinteraksi AWS, Anda menentukan kredensi AWS keamanan Anda untuk memverifikasi siapa Anda dan apakah Anda memiliki izin untuk mengakses sumber daya yang Anda minta. AWS menggunakan kredensi keamanan untuk mengautentikasi dan mengotorisasi permintaan Anda.

Misalnya, jika ingin mengunduh file yang dilindungi dari bucket Amazon Simple Storage Service (Amazon S3), kredensial Anda harus mengizinkan akses tersebut. Jika kredensi Anda tidak menunjukkan bahwa Anda berwenang untuk mengunduh file, tolak permintaan Anda AWS . Namun, kredensi AWS keamanan Anda tidak diperlukan bagi Anda untuk mengunduh file di bucket Amazon S3 yang dibagikan secara publik.

Ada berbagai jenis pengguna di AWS, masing-masing dengan kredensi keamanan mereka sendiri:

- Pemilik akun (pengguna root) — Pengguna yang membuat Akun AWS dan memiliki akses penuh.
- AWS IAM Identity Center pengguna — Pengguna dikelola di AWS IAM Identity Center.
- Pengguna federasi — Pengguna dari penyedia identitas eksternal yang diberikan akses sementara ke AWS melalui federasi. Untuk informasi lebih lanjut tentang identitas federasi, lihat. Penyedia dan federasi identitas
- IAMpengguna — Pengguna individu yang dibuat dalam AWS Identity and Access Management (IAM) layanan.

Pengguna memiliki kredensi keamanan jangka panjang atau sementara. Pengguna root dan IAM pengguna memiliki kredensi keamanan jangka panjang yang tidak kedaluwarsa. Kunci akses adalah kredenal keamanan jangka panjang yang tidak kedaluwarsa. Untuk melindungi kredensi jangka panjang, ada proses untuk mengelola kunci akses, mengubah kata sandi, dan mengaktifkan. MFA Untuk informasi selengkapnya, lihat Praktik terbaik keamanan dan kasus penggunaan AWS Identity and Access Management.

IAMperan, pengguna di Pusat Identitas AWS IAM, dan pengguna federasi memiliki kredensi keamanan sementara. Kredensi keamanan sementara kedaluwarsa setelah jangka waktu tertentu atau ketika pengguna mengakhiri sesi mereka. Kredensi sementara bekerja hampir identik dengan kredensi jangka panjang, dengan perbedaan berikut:

- Kredensial keamanan sementara bersifat jangka pendek, seperti namanya. Konfigurasi dapat berlangsung selama beberapa menit hingga beberapa jam. Setelah kredensialnya kedaluwarsa, AWS tidak lagi mengenalinya atau mengizinkan segala jenis akses dari API permintaan yang dibuat dengannya.
- Kredensial keamanan sementara tidak disimpan dengan pengguna tetapi dihasilkan secara dinamis dan diberikan kepada pengguna saat diminta. Ketika (atau bahkan sebelum) kredensial keamanan sementara kedaluwarsa, pengguna dapat meminta kredensial baru, selama pengguna yang memintanya masih memiliki izin untuk melakukannya.

Akibatnya, kredensi sementara memiliki keunggulan sebagai berikut dibandingkan kredensi jangka panjang:

- Anda tidak perlu mendistribusikan atau menanamkan kredensi AWS keamanan jangka panjang dengan aplikasi.
- Anda dapat memberikan akses ke AWS sumber daya Anda kepada pengguna tanpa harus menentukan AWS identitas untuk mereka. Kredensial sementara adalah dasar untuk peran dan federasi identitas.
- Kredensi keamanan sementara memiliki masa pakai yang terbatas, jadi Anda tidak perlu memperbaruinya atau mencabutnya secara eksplisit saat tidak lagi diperlukan. Setelah kredensial keamanan sementara berakhir, kredensial tersebut tidak dapat digunakan kembali. Anda dapat menentukan berapa lama kredensial berlaku, hingga batas maksimum.

Pertimbangan keamanan

Kami menyarankan Anda mempertimbangkan informasi berikut saat menentukan ketentuan keamanan untuk Anda Akun AWS:

- Saat Anda membuat Akun AWS, kami membuat pengguna root akun. Kredensi pengguna root (pemilik akun) memungkinkan akses penuh ke semua sumber daya di akun. Tugas pertama yang Anda lakukan dengan pengguna root adalah memberikan izin administratif pengguna lain kepada Anda Akun AWS sehingga Anda meminimalkan penggunaan pengguna root.
- Otentikasi multi-faktor (MFA) memberikan tingkat keamanan ekstra bagi pengguna yang dapat mengakses Anda. Akun AWS Untuk keamanan tambahan, kami sarankan Anda meminta MFA Pengguna root akun AWS kredensialnya dan semua IAM pengguna. Untuk informasi selengkapnya, lihat AWS Otentikasi multi-faktor di IAM.

- AWS memerlukan berbagai jenis kredensi keamanan, tergantung pada bagaimana Anda mengakses AWS dan jenis AWS pengguna Anda. Misalnya, Anda menggunakan kredenal masuk untuk AWS Management Console sementara Anda menggunakan kunci akses untuk melakukan panggilan terprogram. AWS Untuk bantuan menentukan jenis pengguna dan halaman login, lihat Apa itu AWS Masuk di AWS Sign-In Panduan Pengguna.
- Anda tidak dapat menggunakan IAM kebijakan untuk menolak akses pengguna root ke sumber daya secara eksplisit. Anda hanya dapat menggunakan kebijakan kontrol AWS Organizations layanan (SCP) untuk membatasi izin pengguna root.
- Jika Anda lupa atau kehilangan kata sandi pengguna root Anda, Anda harus memiliki akses ke alamat email yang terkait dengan akun Anda untuk mengatur ulang.
- Jika Anda kehilangan kunci akses pengguna root Anda, Anda harus dapat masuk ke akun Anda sebagai pengguna root untuk membuat yang baru.
- Jangan gunakan pengguna root untuk tugas sehari-hari Anda. Gunakan untuk melakukan tugas-tugas yang hanya dapat dilakukan oleh pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihatTugas yang memerlukan kredensial pengguna root.
- Kredensial keamanan adalah khusus akun. Jika Anda memiliki akses ke beberapa Akun AWS, Anda memiliki kredensi terpisah untuk setiap akun.
- Kebijakan menentukan tindakan apa yang dapat dilakukan pengguna, peran, atau anggota grup pengguna, pada AWS sumber daya mana, dan dalam kondisi apa. Dengan menggunakan kebijakan, Anda dapat mengontrol akses Layanan AWS dan sumber daya dengan aman di situs Anda Akun AWS. Jika Anda harus mengubah atau mencabut izin sebagai respons terhadap peristiwa keamanan, Anda menghapus atau mengubah kebijakan alih-alih membuat perubahan langsung pada identitas.
- Pastikan untuk menyimpan kredensi masuk untuk IAM pengguna Akses Darurat Anda dan kunci akses apa pun yang Anda buat untuk akses terprogram di lokasi yang aman. Jika Anda kehilangan kunci akses, Anda harus masuk ke akun Anda untuk membuat yang baru.
- Kami sangat menyarankan agar Anda menggunakan kredensi sementara yang disediakan oleh IAM peran dan pengguna federasi, bukan kredenal jangka panjang yang disediakan oleh IAM pengguna dan kunci akses.

Akses terprogram dengan kredensi AWS keamanan

Sebaiknya gunakan kunci akses jangka pendek bila memungkinkan untuk melakukan panggilan terprogram ke AWS atau menggunakan AWS Command Line Interface atau AWS Tools for

PowerShell. Namun, Anda juga dapat menggunakan kunci AWS akses jangka panjang untuk tujuan ini.

Ketika Anda membuat kunci akses jangka panjang, Anda membuat ID kunci akses (misalnya,`AKIAIOSFODNN7EXAMPLE`) dan kunci akses rahasia (misalnya,`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) sebagai satu set. Secret access key hanya tersedia untuk diunduh saat Anda membuatnya. Jika tidak mengunduh secret access key atau kehilangannya, Anda harus membuat yang baru.

Dalam banyak skenario, Anda tidak memerlukan kunci akses jangka panjang yang tidak pernah kedaluwarsa (seperti yang Anda miliki saat membuat kunci akses untuk IAM pengguna). Sebagai gantinya, Anda dapat membuat IAM peran dan menghasilkan kredensi keamanan sementara. Kredensi keamanan sementara termasuk ID kunci akses dan kunci akses rahasia, tetapi mereka juga menyertakan token keamanan yang menunjukkan kapan kredensialnya kedaluwarsa. Setelah kedaluwarsa, mereka tidak lagi valid. Untuk informasi selengkapnya, silakan lihat Alternatif untuk kunci akses jangka panjang

Kunci akses yang IDs dimulai dengan `AKIA` adalah kunci akses jangka panjang untuk IAM pengguna atau pengguna Akun AWS root. Kunci akses yang IDs dimulai dengan `ASIA` adalah kunci akses kredensional sementara yang Anda buat menggunakan AWS STS operasi.

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
Identitas tenaga kerja (Pengguna dikelola di Pusat IAM Identitas)	Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk AWS CLI, lihat Mengkonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
		Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDKs, alat, dan AWS APIs, lihat otentikasi Pusat IAM Identitas di Panduan Referensi Alat AWS SDKs dan Alat.
IAM	Gunakan kredensi sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk dalam Menggunakan kredensi sementara dengan AWS sumber daya di IAMPanduan Pengguna.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
IAM	(Tidak direkomendasikan) Gunakan kredensi jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk mengetahui AWS CLI, lihat Mengautentikasi menggunakan kredensi IAM pengguna di Panduan Pengguna.AWS Command Line Interface • Untuk AWS SDKs dan alat, lihat Mengautentikasi menggunakan kredensi jangka panjang di Panduan Referensi Alat AWS SDKs dan Alat. • Untuk AWS APIs, lihat Mengelola kunci akses untuk IAM pengguna di Panduan IAM Pengguna.

Alternatif untuk kunci akses jangka panjang

Untuk banyak kasus penggunaan umum, ada alternatif untuk kunci akses jangka panjang. Untuk meningkatkan keamanan akun Anda, pertimbangkan hal berikut.

- Jangan menanamkan kunci akses jangka panjang dan kunci akses rahasia dalam kode aplikasi Anda atau dalam repositori kode — Sebagai gantinya, gunakan AWS Secrets Manager, atau solusi manajemen rahasia lainnya, sehingga Anda tidak perlu melakukan hardcode kunci dalam teks biasa. Aplikasi atau klien kemudian dapat mengambil rahasia bila diperlukan. Untuk informasi lebih lanjut, lihat Apa itu AWS Secrets Manager? dalam AWS Secrets Manager User Guide.

- Gunakan IAM peran untuk menghasilkan kredensi keamanan sementara bila memungkinkan — Selalu gunakan mekanisme untuk mengeluarkan kredensi keamanan sementara bila memungkinkan, bukan kunci akses jangka panjang. Kredensi keamanan sementara lebih aman karena tidak disimpan bersama pengguna tetapi dihasilkan secara dinamis dan diberikan kepada pengguna saat diminta. Kredensi keamanan sementara memiliki masa pakai yang terbatas sehingga Anda tidak perlu mengelola atau memperbaruinya. Mekanisme yang menyediakan kunci akses sementara termasuk IAM peran atau otentikasi pengguna Pusat IAM Identitas. Untuk mesin yang berjalan di luar AWS Anda dapat menggunakan AWS Identity and Access Management Peran Di Mana Saja.
- Gunakan alternatif untuk kunci akses jangka panjang untuk AWS Command Line Interface (AWS CLI) atau **`aws-shell`** — Alternatif termasuk yang berikut ini.
  - AWS CloudShelladalah shell berbasis browser dan pra-otentikasi yang dapat Anda luncurkan langsung dari file. AWS Management Console Anda dapat menjalankan AWS CLI perintah melawan Layanan AWS melalui shell pilihan Anda (Bash, Powershell, atau Z shell). Ketika Anda melakukan ini, Anda tidak perlu mengunduh atau menginstal alat baris perintah. Untuk informasi selengkapnya, lihat Apa itu AWS CloudShell? dalam AWS CloudShell Panduan Pengguna.
  - AWS CLI Integrasi versi 2 dengan AWS IAM Identity Center (Pusat IAM Identitas). Anda dapat mengautentikasi pengguna dan memberikan kredensi jangka pendek untuk menjalankan perintah. AWS CLI Untuk mempelajari lebih lanjut, lihat Mengintegrasikan AWS CLI dengan Pusat IAM Identitas di Panduan AWS IAM Identity Center Pengguna dan Mengonfigurasi Pusat IAM Identitas AWS CLI untuk menggunakan di Panduan AWS Command Line Interface Pengguna.
- Jangan membuat kunci akses jangka panjang untuk pengguna manusia yang memerlukan akses ke aplikasi atau Layanan AWS — Pusat IAM Identitas dapat menghasilkan kredenal akses sementara untuk diakses oleh pengguna iDP eksternal Anda. Layanan AWS Ini menghilangkan kebutuhan untuk membuat dan mengelola kredensi jangka panjang di. IAM Di Pusat IAM Identitas, buat set izin Pusat IAM Identitas yang memberikan akses pengguna iDP eksternal. Kemudian tetapkan grup dari Pusat IAM Identitas ke set izin di yang dipilih Akun AWS. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center, Connect ke penyedia identitas eksternal Anda, dan set Izin di Panduan AWS IAM Identity Center Pengguna.
- Jangan menyimpan kunci akses jangka panjang dalam layanan AWS komputasi — Sebagai gantinya, tetapkan IAM peran untuk menghitung sumber daya. Ini secara otomatis memasok kredenal sementara untuk memberikan akses. Misalnya, saat membuat profil instans yang dilampirkan ke EC2 instans Amazon, Anda dapat menetapkan AWS peran ke instance tersebut dan membuatnya tersedia untuk semua aplikasinya. Profil instans berisi peran dan memungkinkan

program yang berjalan di EC2 instans Amazon untuk mendapatkan kredensi sementara. Untuk mempelajari lebih lanjut, lihat Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon.

AWS pedoman audit keamanan

Audit konfigurasi keamanan Anda secara berkala untuk memastikannya memenuhi kebutuhan bisnis Anda saat ini. Audit memberi Anda kesempatan untuk menghapus IAM pengguna, peran, grup, dan kebijakan yang tidak diperlukan, dan untuk memastikan bahwa pengguna dan perangkat lunak Anda tidak memiliki izin yang berlebihan.

Berikut ini adalah pedoman untuk meninjau dan memantau AWS sumber daya Anda secara sistematis untuk praktik terbaik keamanan.

> **Tip**
>
> Anda dapat memantau penggunaan Anda IAM karena berkaitan dengan praktik terbaik keamanan dengan menggunakan AWS Security Hub. Hub Keamanan menggunakan kontrol keamanan untuk mengevaluasi konfigurasi sumber daya dan standar keamanan guna membantu Anda mematuhi berbagai kerangka kerja kepatuhan. Untuk informasi selengkapnya tentang penggunaan Security Hub guna mengevaluasi IAM sumber daya, lihat kontrol AWS Identity and Access Management di Panduan AWS Security Hub Pengguna.

Daftar Isi

Kapan melakukan audit keamanan

Audit konfigurasi keamanan Anda dalam situasi berikut:

- Secara periodik. Lakukan langkah-langkah yang dijelaskan dalam dokumen ini secara berkala sebagai praktik terbaik untuk keamanan.
- Jika ada perubahan di organisasi Anda, seperti orang yang keluar.
- Jika Anda telah berhenti menggunakan satu atau beberapa AWS layanan individual untuk memverifikasi bahwa Anda telah menghapus izin yang tidak lagi diperlukan oleh pengguna di akun Anda.
- Jika Anda telah menambahkan atau menghapus perangkat lunak di akun Anda, seperti aplikasi di EC2 instans Amazon, AWS OpsWorks tumpukan, AWS CloudFormation templat, dll.
- Jika Anda mencurigai bahwa orang yang tidak berwenang mungkin telah mengakses akun Anda.

Pedoman untuk audit

Saat meninjau konfigurasi keamanan akun Anda, ikuti panduan berikut:

- Jadilah teliti. Lihatlah semua aspek konfigurasi keamanan Anda, termasuk yang jarang digunakan.
- Jangan berasumsi. Jika Anda tidak terbiasa dengan beberapa aspek konfigurasi keamanan Anda (misalnya, alasan di balik kebijakan tertentu atau keberadaan peran), selidiki kebutuhan bisnis sampai Anda memahami potensi risikonya.
- Tetap sederhana. Untuk mempermudah audit (dan manajemen), gunakan IAM grup, IAM peran, skema penamaan yang konsisten, dan kebijakan langsung.

Tinjau kredensi AWS akun Anda

Ambil langkah-langkah ini saat Anda mengaudit kredensi AWS akun Anda:

1. Jika Anda memiliki kunci akses untuk pengguna root yang tidak Anda gunakan, Anda dapat menghapusnya. Kami sangat menyarankan agar Anda tidak menggunakan kunci akses root untuk pekerjaan sehari-hari AWS, dan sebaliknya Anda menggunakan pengguna dengan kredensi sementara, seperti itu. pengguna di Pusat Identitas AWS IAM
2. Jika Anda memerlukan kunci akses untuk akun Anda, pastikan Anda memperbaruinya saat diperlukan.

Tinjau IAM pengguna Anda

Ambil langkah-langkah ini saat Anda mengaudit IAM pengguna yang sudah ada:

1. Buat daftar pengguna Anda dan kemudian hapus pengguna yang tidak diperlukan.
2. Hapus pengguna dari grup yang tidak mereka perlukan aksesnya.
3. Tinjau kebijakan yang dilampirkan ke grup tempat pengguna berada. Lihat Kiat untuk meninjau kebijakan IAM.
4. Hapus kredensial keamanan yang tidak dibutuhkan pengguna atau yang mungkin telah diekspos. Misalnya, IAM pengguna yang digunakan untuk aplikasi tidak memerlukan kata sandi (yang diperlukan hanya untuk masuk ke AWS situs web). Demikian pula, jika pengguna tidak menggunakan kunci akses, tidak ada alasan bagi pengguna untuk memilikinya. Untuk informasi selengkapnya, lihat Mengelola Kata Sandi untuk IAM pengguna dan Mengelola Kunci Akses untuk IAM pengguna.

   Anda dapat membuat dan mengunduh laporan kredensi yang mencantumkan semua IAM pengguna di akun Anda dan status berbagai kredensialnya, termasuk kata sandi, kunci akses, dan perangkat. MFA Untuk kata sandi dan kunci akses, laporan kredensi menunjukkan tanggal dan waktu ketika kata sandi atau kunci akses terakhir digunakan. Pertimbangkan untuk menghapus kredensil yang belum digunakan baru-baru ini dari akun Anda. (Jangan hapus pengguna Akses Darurat Anda.) Untuk informasi selengkapnya, lihat Mendapatkan Laporan Kredensi untuk AWS Akun Anda.
5. Perbarui kata sandi dan kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredensi jangka panjang. Untuk informasi selengkapnya, lihat Mengelola Kata Sandi untuk IAM Pengguna dan Mengelola Kunci Akses untuk IAM pengguna.
6. Sebagai praktik terbaik, mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara. Jika memungkinkan, transisi dari IAM pengguna ke pengguna federasi, seperti pengguna di Pusat IAM Identitas. Pertahankan jumlah minimum IAM pengguna yang dibutuhkan untuk aplikasi Anda.

Tinjau IAM grup Anda

Ambil langkah-langkah ini saat Anda mengaudit IAM grup Anda:

1. Buat daftar grup Anda, lalu hapus grup yang tidak Anda gunakan.
2. Tinjau pengguna di setiap grup dan hapus pengguna yang tidak termasuk.

3. Tinjau kebijakan yang dilampirkan ke grup. Lihat Kiat untuk meninjau kebijakan IAM.

Tinjau IAM peran Anda

Ambil langkah-langkah ini saat Anda mengaudit IAM peran Anda:

1. Buat daftar peran Anda, lalu hapus peran yang tidak Anda gunakan.
2. Tinjau kebijakan kepercayaan peran. Pastikan Anda mengetahui siapa akun utamanya dan memahami mengapa akun atau pengguna tersebut harus dapat mengambil peran tersebut.
3. Tinjau kebijakan akses untuk peran tersebut guna memastikan bahwa kebijakan akses memberikan izin yang sesuai kepada siapa pun yang mengambil peran tersebut—lihat Kiat untuk meninjau kebijakan IAM.

Tinjau IAM penyedia Anda untuk SAML dan OpenID Connect () OIDC

Jika Anda telah membuat IAM entitas untuk membangun kepercayaan dengan SAMLatau penyedia OIDC identitas (iDP), lakukan langkah-langkah berikut:

1. Hapus penyedia yang tidak digunakan.
2. Unduh dan tinjau dokumen AWS metadata untuk setiap SAML IDP dan pastikan dokumen tersebut mencerminkan kebutuhan bisnis Anda saat ini.
3. Dapatkan dokumen metadata terbaru dari SAML IdPs dan perbarui penyedia di. IAM

Tinjau aplikasi seluler Anda

Jika Anda telah membuat aplikasi seluler yang membuat permintaan AWS, lakukan langkah-langkah berikut:

1. Pastikan aplikasi seluler tidak berisi kunci akses yang disematkan, meskipun berada di penyimpanan terenkripsi.
2. Dapatkan kredensi sementara untuk aplikasi dengan menggunakan yang APIs dirancang untuk tujuan itu.

> **Note**
>
> Sebaiknya gunakan Amazon Cognito untuk mengelola identitas pengguna di aplikasi Anda. Layanan ini memungkinkan Anda mengautentikasi pengguna menggunakan Login with Amazon, Facebook, Google, atau penyedia identitas yang kompatibel dengan OpenID OIDC Connect (). Untuk informasi selengkapnya, lihat kumpulan identitas Amazon Cognito di Panduan Pengembang Amazon Cognito.

Kiat untuk meninjau kebijakan IAM

Kebijakan sangat kuat dan halus, jadi penting untuk mempelajari dan memahami izin yang diberikan oleh setiap kebijakan. Gunakan pedoman berikut saat meninjau kebijakan:

- Lampirkan kebijakan ke grup atau peran, bukan ke pengguna individu. Jika pengguna individu memiliki kebijakan, pastikan Anda memahami mengapa pengguna tersebut memerlukan kebijakan tersebut.
- Pastikan bahwa IAM pengguna, grup, dan peran memiliki izin yang mereka butuhkan dan tidak memiliki izin tambahan.
- Gunakan Simulator IAM Kebijakan untuk menguji kebijakan yang dilampirkan ke pengguna atau grup.
- Ingatlah bahwa izin pengguna adalah hasil dari semua kebijakan yang berlaku — baik kebijakan berbasis identitas (pada pengguna, grup, atau peran) maupun kebijakan berbasis sumber daya (pada sumber daya seperti bucket Amazon S3, antrian Amazon, topik Amazon, dan kunci). SQS SNS AWS KMS Penting untuk memeriksa semua kebijakan yang berlaku bagi pengguna dan untuk memahami set lengkap izin yang diberikan kepada pengguna individu.
- Ketahuilah bahwa mengizinkan pengguna untuk membuat IAM pengguna, grup, peran, atau kebijakan dan melampirkan kebijakan ke entitas utama secara efektif memberikan pengguna tersebut semua izin ke semua sumber daya di akun Anda. Pengguna yang dapat membuat kebijakan dan melampirkannya ke pengguna, grup, atau peran dapat memberikan izin apa pun kepada diri mereka sendiri. Secara umum, jangan memberikan IAM izin kepada pengguna atau peran yang tidak Anda percayai dengan akses penuh ke sumber daya di akun Anda. Saat melakukan audit keamanan, konfirmasikan bahwa IAM izin berikut diberikan kepada identitas tepercaya:
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`

  - `iam:PutUserPolicy`
  - `iam:CreatePolicy`
  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- Pastikan kebijakan tidak memberikan izin untuk layanan yang tidak Anda gunakan. Misalnya, jika Anda menggunakan kebijakan AWS terkelola, pastikan kebijakan AWS terkelola yang digunakan di akun Anda adalah untuk layanan yang benar-benar Anda gunakan. Untuk mengetahui kebijakan AWS terkelola mana yang digunakan di akun Anda, gunakan IAM `GetAccountAuthorizationDetails`API(AWS CLI perintah: `aws iam get-account-authorization-details`).
- Jika kebijakan memberikan izin kepada pengguna untuk meluncurkan EC2 instance Amazon, kebijakan tersebut mungkin juga mengizinkan `iam:PassRole` tindakan tersebut, tetapi jika demikian, kebijakan tersebut harus secara eksplisit mencantumkan peran yang dapat diteruskan pengguna ke instance Amazon. EC2
- Memeriksa setiap nilai untuk `Action` atau `Resource` elemen yang termasuk`*`. Jika memungkinkan, berikan `Allow` akses ke tindakan individu dan sumber daya yang dibutuhkan pengguna. Namun, berikut ini adalah alasan yang mungkin cocok untuk menggunakan * dalam kebijakan:
  - Kebijakan ini dirancang untuk memberikan izin tingkat administratif.
  - Karakter wildcard digunakan untuk set tindakan serupa (misalnya, `Describe*`) sebagai kenyamanan dan Anda merasa nyaman dengan daftar lengkap tindakan yang direferensikan dengan cara ini.
  - Karakter wildcard digunakan untuk menunjukkan kelas sumber daya atau jalur sumber daya (misalnya,`arn:aws:iam::`*`account-id`*`:users/division_abc/*`), dan Anda merasa nyaman memberikan akses ke semua sumber daya di kelas atau jalur itu.
  - Tindakan layanan tidak mendukung izin tingkat sumber daya, dan satu-satunya pilihan untuk sumber daya adalah. *
- Periksa nama kebijakan untuk memastikan mereka mencerminkan fungsi kebijakan. Misalnya, meskipun kebijakan mungkin memiliki nama yang menyertakan "hanya baca", kebijakan tersebut mungkin benar-benar memberikan izin tulis atau ubah.

Untuk informasi selengkapnya tentang perencanaan audit keamanan Anda, lihat Praktik Terbaik untuk Keamanan, Identitas, dan Kepatuhan di Pusat AWS Arsitektur.

Perlindungan data di AWS Identity and Access Management

Bagian AWS model tanggung jawab bersama model berlaku untuk perlindungan data di AWS Identity and Access Management. Seperti yang dijelaskan dalam model ini, AWS bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk menjaga kontrol atas konten Anda yang di-host di infrastruktur ini. Anda juga bertanggung jawab atas konfigurasi keamanan dan tugas manajemen untuk Layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, lihat Privasi Data FAQ. Untuk informasi tentang perlindungan data di Eropa, lihat AWS Model Tanggung Jawab Bersama dan posting GDPR blog di AWS Blog Keamanan.

Untuk tujuan perlindungan data, kami menyarankan Anda untuk melindungi Akun AWS kredensi dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan otentikasi multi-faktor (MFA) dengan setiap akun.
- GunakanSSL/TLSuntuk berkomunikasi dengan AWS sumber daya. Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang menggunakan CloudTrail jalur untuk menangkap AWS kegiatan, lihat Bekerja dengan CloudTrail jalan setapak di AWS CloudTrail Panduan Pengguna.
- Gunakan AWS solusi enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan FIPS 140-3 modul kriptografi yang divalidasi saat mengakses AWS melalui antarmuka baris perintah atauAPI, gunakan FIPS titik akhir. Untuk informasi selengkapnya tentang FIPS titik akhir yang tersedia, lihat Federal Information Processing Standard (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas

seperti bidang Nama. Ini termasuk ketika Anda bekerja dengan IAM atau lainnya Layanan AWS menggunakan konsol,API, AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Jika Anda memberikan URL ke server eksternal, kami sangat menyarankan agar Anda tidak menyertakan informasi kredensil dalam URL untuk memvalidasi permintaan Anda ke server tersebut.

Enkripsi data di IAM dan AWS STS

Enkripsi data biasanya terbagi dalam dua kategori: enkripsi saat diam dan enkripsi dalam transit.

Enkripsi diam

Data yang dikumpulkan dan disimpan oleh IAM dienkripsi saat istirahat.

- IAMData yang dikumpulkan dan disimpan di dalamnya IAM termasuk alamat IP, metadata akun pelanggan, dan data identifikasi pelanggan yang menyertakan kata sandi. Metadata akun pelanggan dan data identifikasi pelanggan dienkripsi saat istirahat menggunakan AES 256 atau di-hash menggunakan 256. SHA
- AWS STS – AWS STS tidak mengumpulkan konten pelanggan kecuali untuk log layanan yang mencatat permintaan yang berhasil, salah, dan salah ke layanan.

Enkripsi bergerak

Data identifikasi pelanggan, termasuk kata sandi, dienkripsi dalam perjalanan menggunakan TLS 1.2 dan 1.3. Semua AWS STS dukungan endpoint HTTPS untuk mengenkripsi data dalam perjalanan. Untuk daftar AWS STS titik akhir, lihatAWS STS Wilayah dan titik akhir.

Manajemen kunci di IAM dan AWS STS

Anda tidak dapat mengelola kunci enkripsi menggunakan IAM atau AWS STS. Untuk informasi selengkapnya tentang kunci enkripsi, lihat Apa itu AWS KMS? di AWS Key Management Service Panduan Developer

Privasi lalu lintas internetwork di dan IAM AWS STS

Permintaan IAM harus dibuat menggunakan protokol Transport Layer Security (TLS). Anda dapat mengamankan koneksi ke AWS STS layanan dengan menggunakan VPC titik akhir. Untuk mempelajari informasi lebih lanjut, lihat Titik VPC akhir antarmuka.

Penebangan dan pemantauan di AWS Identity and Access Management

Pemantauan adalah bagian penting dari menjaga keandalan, ketersediaan, dan kinerja AWS Identity and Access Management (IAM), AWS Security Token Service (AWS STS) dan AWS solusi Anda yang lain. AWS menyediakan beberapa alat untuk memantau AWS sumber daya Anda dan menanggapi potensi insiden:

- AWS CloudTrailmenangkap semua API panggilan untuk IAM dan AWS STS sebagai acara, termasuk panggilan dari konsol dan API panggilan. Untuk mempelajari lebih lanjut tentang menggunakan CloudTrail dengan IAM dan AWS STS, lihatPenebangan IAM dan AWS STS APIpanggilan dengan AWS CloudTrail. Untuk informasi selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.
- AWS Identity and Access Management Access Analyzer membantu Anda mengidentifikasi sumber daya di organisasi dan akun Anda, seperti bucket IAM atau peran Amazon S3, yang dibagikan dengan entitas eksternal. Hal ini membantu Anda mengidentifikasi akses yang tidak diinginkan ke sumber daya dan data Anda, yang merupakan risiko keamanan. Untuk mempelajari lebih lanjut, lihat Apa itu IAM Access Analyzer?
- Amazon CloudWatch memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. Anda dapat mengumpulkan dan melacak metrik, membuat dasbor yang disesuaikan, dan mengatur alarm yang memberi tahu Anda atau mengambil tindakan saat metrik tertentu mencapai ambang batas yang ditentukan. Misalnya, Anda dapat CloudWatch melacak CPU penggunaan atau metrik lain dari EC2 instans Amazon Anda dan secara otomatis meluncurkan instans baru bila diperlukan. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.
- Amazon CloudWatch Logs membantu Anda memantau, menyimpan, dan mengakses file log Anda dari EC2 instans Amazon CloudTrail, dan sumber lainnya. CloudWatch Log dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang batas tertentu terpenuhi. Anda juga dapat mengarsipkan data log dalam penyimpanan yang sangat durabel. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon CloudWatch Logs.

Untuk sumber daya tambahan dan praktik terbaik keamananIAM, lihatPraktik terbaik keamanan dan kasus penggunaan AWS Identity and Access Management.

Topik

Penebangan IAM dan AWS STS APIpanggilan dengan AWS CloudTrail

IAMdan AWS STS terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh IAM pengguna atau peran. CloudTrail menangkap semua API panggilan untuk IAM dan AWS STS sebagai acara, termasuk panggilan dari konsol dan dari API panggilan. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara terus menerus ke bucket Amazon S3. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Anda dapat menggunakan CloudTrail untuk mendapatkan informasi tentang permintaan yang dibuat untuk IAM atau AWS STS. Misalnya, Anda dapat melihat alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari lebih lanjut tentang CloudTrail, lihat AWS CloudTrail Panduan Pengguna.

Topik

IAMdan AWS STS informasi di CloudTrail

CloudTrail diaktifkan pada Akun AWS saat Anda membuat akun. Ketika aktivitas terjadi di IAM atau AWS STS, kegiatan tersebut dicatat dalam suatu CloudTrail peristiwa bersama dengan yang lain AWS acara layanan dalam sejarah Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di Akun AWS. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di Akun AWS, termasuk acara untuk IAM dan AWS STS, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol, jejak diterapkan ke semua Region. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengkonfigurasi AWS layanan untuk menganalisis lebih lanjut dan bertindak atas data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengkonfigurasi SNS Pemberitahuan Amazon untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Semua IAM dan AWS STS tindakan dicatat oleh CloudTrail dan didokumentasikan dalam IAMAPIReferensi dan AWS Security Token Service APIReferensi.

Penebangan IAM dan AWS STS APIpermintaan

CloudTrail mencatat semua API permintaan yang diautentikasi ke dan IAM AWS STS APIoperasi. CloudTrail juga mencatat permintaan yang tidak diautentikasi ke AWS STS tindakan, `AssumeRoleWithSAML` dan`AssumeRoleWithWebIdentity`, dan mencatat informasi yang disediakan oleh penyedia identitas. Namun, beberapa yang tidak diautentikasi AWS STS permintaan mungkin tidak dicatat karena mereka tidak memenuhi harapan minimum yang cukup valid untuk dipercaya sebagai permintaan yang sah.

Anda dapat menggunakan informasi yang dicatat untuk memetakan panggilan yang dilakukan oleh pengguna federasi dengan peran yang diasumsikan kembali ke pemanggil federasi eksternal yang berasal. Dalam hal ini`AssumeRole`, Anda dapat memetakan panggilan kembali ke aslinya AWS layanan atau ke akun pengguna asal. `userIdentity`Bagian JSON data dalam entri CloudTrail log berisi informasi yang Anda butuhkan untuk memetakan AssumeRole* permintaan dengan pengguna federasi tertentu. Untuk informasi selengkapnya, lihat CloudTrail userIdentityElemen di AWS CloudTrail Panduan Pengguna.

Misalnya, panggilan ke IAM`CreateUser`,, `DeleteRole``ListGroups`, dan API operasi lainnya semuanya dicatat oleh CloudTrail.

Contoh untuk jenis entri log ini akan disampaikan nanti dalam topik ini.

APIPermintaan pencatatan ke orang lain AWS layanan

Permintaan yang diautentikasi ke orang lain AWS APIoperasi layanan dicatat oleh CloudTrail, dan entri log ini berisi informasi tentang siapa yang membuat permintaan.

Misalnya, anggap Anda membuat permintaan untuk mencantumkan EC2 instans Amazon atau membuat AWS CodeDeploy kelompok penyebaran. Perincian tentang orang atau layanan yang mengajukan permintaan tersebut terdapat di entri log untuk permintaan tersebut. Informasi ini membantu Anda menentukan apakah permintaan tersebut dibuat oleh Pengguna root akun AWS, IAM pengguna, peran, atau lainnya AWS layanan.

Untuk detail selengkapnya tentang informasi identitas pengguna di entri CloudTrail log, lihat userIdentity Elemen di AWS CloudTrail Panduan Pengguna.

Mencatat peristiwa masuk pengguna

CloudTrail mencatat peristiwa login ke AWS Management Console, AWS forum diskusi, dan AWS Marketplace. CloudTrailmencatat upaya masuk yang berhasil dan gagal untuk IAM pengguna dan pengguna gabungan.

Untuk melihat contoh CloudTrail peristiwa untuk login pengguna root yang berhasil dan tidak berhasil, lihat Contoh catatan peristiwa untuk pengguna root di AWS CloudTrail Panduan Pengguna.

Sebagai praktik keamanan terbaik, AWS tidak mencatat teks nama IAM pengguna yang dimasukkan saat kegagalan masuk disebabkan oleh nama pengguna yang salah. Teks nama pengguna ditutupi oleh nilai `HIDDEN_DUE_TO_SECURITY_REASONS`. Untuk contoh ini, lihat Contoh peristiwa gagal masuk yang disebabkan oleh nama pengguna yang salah, nanti dalam topik ini. Teks nama pengguna disamarkan karena kegagalan tersebut mungkin disebabkan oleh kesalahan pengguna. Pembuatan catatan kesalahan ini dapat memaparkan informasi yang berpotensi sensitif. Sebagai contoh:

- Anda secara tidak sengaja memasukkan kata sandi anda di kotak nama pengguna.
- Anda memilih tautan untuk halaman masuk salah satunya Akun AWS, tetapi kemudian ketik nomor akun untuk yang berbeda Akun AWS.
- Anda lupa akun mana yang sedang Anda masuki dan secara tidak sengaja mengetikkan nama akun dari akun email pribadi Anda, pengenal masuk bank Anda, atau beberapa ID pribadi lainnya.

Mencatat peristiwa masuk untuk kredensial sementara

Ketika prinsipal meminta kredensi sementara, tipe utama menentukan cara CloudTrail mencatat peristiwa. Ini bisa jadi rumit ketika pelaku utama mengambil peran dalam akun lain. Ada beberapa API panggilan untuk melakukan operasi yang terkait dengan operasi lintas akun peran. Pertama, kepala sekolah memanggil AWS STS APIuntuk mengambil kredensi sementara. Operasi itu masuk ke akun panggilan dan akun tempat AWS STS operasi dilakukan. Kemudian kepala sekolah kemudian menggunakan peran tersebut untuk melakukan API panggilan lain di akun peran yang diasumsikan.

Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan identias saat mereka mengasumsikan sebuah peran. Misalnya, Anda dapat meminta IAM pengguna menentukan nama pengguna mereka sendiri sebagai identitas sumber mereka. Ini dapat membantu Anda menentukan pengguna mana yang melakukan tindakan tertentu di AWS Untuk informasi selengkapnya, lihat `sts:SourceIdentity`. Anda dapat menggunakan `sts:RoleSessionName` untuk mengharuskan pengguna menentukan nama sesi saat mereka mengasumsikan sebuah peran. Ini dapat membantu Anda membedakan antara sesi peran untuk peran yang digunakan oleh kepala sekolah yang berbeda saat Anda meninjau AWS CloudTrail log.

Tabel berikut menunjukkan bagaimana CloudTrail log informasi identitas pengguna yang berbeda untuk masing-masing AWS STS APIsyang menghasilkan kredensi sementara.

Tipe utama	STS API	Identitas pengguna di CloudTrail log untuk akun penelepon	Identitas pengguna di CloudTrail log untuk akun peran yang diasumsikan	Identitas pengguna di CloudTrail log untuk API panggilan peran berikutnya
Pengguna root akun AWS credentials	GetSessionToken	Identitas pengguna root	Akun pemilik peran sama dengan akun panggilan	Identitas pengguna root
IAMpengguna	GetSessionToken	Identitas pengguna IAM	Akun pemilik peran sama	Identitas pengguna IAM

Tipe utama	STS API	Identitas pengguna di CloudTrail log untuk akun penelepon	Identitas pengguna di CloudTrail log untuk akun peran yang diasumsikan	Identitas pengguna di CloudTrail log untuk API panggilan peran berikutnya
			dengan akun panggilan	
IAMpengguna	GetFederationToken	Identitas pengguna IAM	Akun pemilik peran sama dengan akun panggilan	Identitas pengguna IAM
IAMpengguna	AssumeRole	Identitas pengguna IAM	Nomor akun dan ID utama (jika pengguna), atau AWS layanan utama	Khusus identitas peran (tanpa pengguna)
Pengguna yang diautentikasi secara eksternal	AssumeRoleWithSAML	T/A	Identitas pengguna SAML	Khusus identitas peran (tanpa pengguna)
Pengguna yang diautentikasi secara eksternal	AssumeRoleWithWebIdentity	T/A	OIDC/Identitas pengguna web	Khusus identitas peran (tanpa pengguna)

CloudTrail menganggap tindakan hanya-baca jika tidak memiliki efek mutasi pada sumber daya. Saat mencatat peristiwa hanya-baca, CloudTrail menyunting `responseElements` informasi di log. Saat CloudTrail mencatat peristiwa yang tidak hanya-baca, lengkap `responseElements` ditampilkan di entri log. Namun, untuk AWS STS APIs`AssumeRole`,`AssumeRoleWithSAML`, dan`AssumeRoleWithWebIdentity`, meskipun mereka dicatat sebagai hanya-baca, CloudTrail akan menyertakan lengkap `responseElements` dalam log untuk ini. APIs

Tabel berikut menunjukkan bagaimana CloudTrail log `responseElements` dan `readOnly` informasi untuk masing-masing AWS STS APIsyang menghasilkan kredensi sementara.

STS API	Informasi elemen respons	Hanya baca
AssumeRole	Termasuk	true
AssumeRoleWithSAML	Termasuk	true
AssumeRoleWithWebIdentity	Termasuk	true
GetFederationToken	Termasuk	false
GetSessionToken	Termasuk	false

Contoh IAM API peristiwa di CloudTrail log

CloudTrail file log berisi peristiwa yang diformat menggunakanJSON. APIPeristiwa mewakili satu API permintaan dan mencakup informasi tentang prinsipal, tindakan yang diminta, parameter apa pun, dan tanggal dan waktu tindakan.

Contoh IAM API peristiwa dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk IAM `GetUserPolicy` tindakan tersebut.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/JaneDoe",
    "accountId": "444455556666",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "userName": "JaneDoe",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2014-07-15T21:39:40Z"
      }
    },
    "invokedBy": "signin.amazonaws.com"
  },
  "eventTime": "2014-07-15T21:40:14Z",
```

```
  "eventSource": "iam.amazonaws.com",
  "eventName": "GetUserPolicy",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "signin.amazonaws.com",
  "userAgent": "signin.amazonaws.com",
  "requestParameters": {
    "userName": "JaneDoe",
    "policyName": "ReadOnlyAccess-JaneDoe-201407151307"
  },
  "responseElements": null,
  "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
  "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"
}
```

Dari informasi peristiwa ini, Anda dapat menentukan bahwa permintaan dibuat untuk mendapatkan kebijakan pengguna bernama `ReadOnlyAccess-JaneDoe-201407151307` untuk pengguna `JaneDoe`, sebagaimana ditentukan dalam elemen `requestParameters`. Anda juga dapat melihat bahwa permintaan tersebut dibuat oleh IAM pengguna bernama `JaneDoe` pada 15 Juli 2014 pukul 21:40 (UTC). Dalam hal ini, permintaan berasal dari AWS Management Console, seperti yang Anda tahu dari `userAgent` elemen.

Contoh AWS STS APIperistiwa di CloudTrail log

CloudTrail file log berisi peristiwa yang diformat menggunakanJSON. APIPeristiwa mewakili satu API permintaan dan mencakup informasi tentang prinsipal, tindakan yang diminta, parameter apa pun, dan tanggal dan waktu tindakan.

Contoh lintas akun AWS STS APIperistiwa dalam file CloudTrail log

IAMPengguna yang disebutkan `JohnDoe` dalam akun 777788889999 memanggil AWS STS AssumeRoletindakan untuk mengambil peran `EC2-dev` dalam akun 111122223333. Administrator akun mengharuskan pengguna untuk menetapkan identitas sumber yang sama dengan nama pengguna mereka ketika mengambil peran. Pengguna melewati nilai identitas sumber `JohnDoe`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/JohnDoe",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    },
  "sourceIdentity": "JohnDoe"
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Contoh kedua menunjukkan entri CloudTrail log akun peran yang diasumsikan (111122223333) untuk permintaan yang sama.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
      },
  "sourceIdentity": "JohnDoe"
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```


Contoh AWS STS APIperistiwa rantai peran dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat oleh John Doe di akun 111111111111. John sebelumnya menggunakan pengguna `JohnDoe` untuk mengambil peran `JohnRole1`. Untuk permintaan ini, dia menggunakan kredensial dari peran tersebut untuk

mengambil peran `JohnRole2`. Ini dikenal sebagai perangkaian peran. Identitas sumber yang ia tetapkan ketika ia diasumsikan peran `JohnDoe1` tetap ada dalam permintaan untuk mengasumsikan `JohnRole2`. Jika John mencoba untuk menetapkan identitas sumber yang berbeda ketika mengambil peran, permintaan ditolak. John memberikan dua tanda sesi ke dalam permintaan. Dia menetapkan kedua tanda itu sebagai transitif. Permintaan mewarisi tanda `Department` sebagai transitif karena John mengaturnya sebagai transitif ketika ia mengasumsikan `JohnRole1`. Untuk informasi lebih lanjut tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan. Untuk informasi lebih lanjut tentang kunci transitif dalam rantai peran, lihat Merangkai peran dengan tag sesi.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/JohnDoe/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
            },
                "sessionIssuer": {
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "JohnDoe"
            },
            "sourceIdentity": "JohnDoe"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
 Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
```

```
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "JohnDoe",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
            "expiration": "Oct 2, 2019, 11:12:29 PM",
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
    "sourceIdentity": "JohnDoe"
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
```

```
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```

Contoh AWS layanan AWS STS APIperistiwa dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat oleh AWS layanan memanggil layanan lain API menggunakan izin dari peran layanan. Ini menunjukkan entri CloudTrail log untuk permintaan yang dibuat di akun 777788889999.

```
{
  "eventVersion": "1.04",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
```

```
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "amzn-s3-demo-bucket"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```

Contoh SAML AWS STS APIperistiwa dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk AWS STS AssumeRoleWithSAMLtindakan. Permintaan mencakup SAML atribut `CostCenter` dan `Project` yang diteruskan melalui SAML pernyataan sebagai tag sesi. Tanda itu diatur sebagai transitif sehingga mereka bertahan dalam skenario perangkaian peran. Permintaan termasuk API parameter opsional`DurationSeconds`, direpresentasikan seperti `durationSeconds` dalam CloudTrail log, dan diatur ke `1800` detik. Permintaan juga mencakup SAML atribut`sourceIdentity`, yang diteruskan dalam SAML pernyataan. Jika seseorang menggunakan kredensial sesi peran yang dihasilkan untuk mengambil peran lain, identitas sumber ini tetap ada.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
    "eventTime": "2023-08-28T18:30:58Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
```

```
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 1,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
    },
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
```

```
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```

Contoh OIDC AWS STS APIperistiwa dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk AWS STS AssumeRoleWithWebIdentitytindakan. Permintaan menyertakan atribut `CostCenter` dan `Project` yang diteruskan melalui token penyedia identitas OpenID Connect (OIDC) (iDP) sebagai tag sesi. Tanda itu diatur sebagai transitif sehingga mereka bertahan dalam perangkaian peran. Permintaan tersebut mencakup atribut `sourceIdentity` dari token penyedia identitas. Jika seseorang menggunakan kredensial sesi peran yang dihasilkan untuk mengambil peran lain, identitas sumber ini tetap ada.

Entri CloudTrail log juga berisi `additionalEventData` bidang dengan `identityProviderConnectionVerificationMethod` atribut. Atribut ini menunjukkan metode AWS digunakan untuk memverifikasi koneksi dengan OIDC penyedia. Nilai atribut akan menjadi salah satu `IAMTrustStore` atau`Thumbprint`. `IAMTrustStore`Nilai tersebut menunjukkan bahwa AWS berhasil memverifikasi koneksi dengan OIDC IDP menggunakan pustaka otoritas sertifikat root tepercaya kami ()CAs. `Thumbprint`Nilai tersebut menunjukkan bahwa AWS menggunakan cap jempol sertifikat yang diatur dalam konfigurasi iDP untuk memverifikasi sertifikat server OIDC iDP.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>:<id of application>:<id of user>",
```

```
    "userName": "<id of user>",
    "identityProvider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>"
  },
  "eventTime": "2024-07-09T15:41:37Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/2.13.29 Python/3.11.6 Windows/10 exe/AMD64 prompt/off command/
sts.assume-role-with-web-identity",
  "requestParameters": {
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "<assigned role session name>",
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "principalTags": {
      "CostCenter": "24680",
      "Project": "Pegasus"
    },
    "transitiveTagKeys": [
      "CostCenter",
      "Project"
    ]
  },
  "responseElements": {
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "sessionToken": "<encoded session token blob>",
      "expiration": "Jul 9, 2024, 4:41:37 PM"
    },
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "assumedRoleUser": {
      "assumedRoleId": "AROA123456789EXAMPLE:<assigned role session name>",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/<assigned
 role session name>"
    },
    "provider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>",
    "audience": "<id of application>"
  },
  "additionalEventData": {
    "identityProviderConnectionVerificationMethod": "IAMTrustStore"
```

```
  },
  "requestID": "aEXAMPLE-0b26-40df-8973-c7012EXAMPLE",
  "eventID": "aEXAMPLE-ee29-4ac0-a0ed-3f5c5EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "444455556666",
      "type": "AWS::IAM::Role",
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "444455556666",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_128_GCM_SHA256",
    "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
  }
}
```

Contoh peristiwa masuk dalam catatan CloudTrail

CloudTrail file log berisi peristiwa yang diformat menggunakanJSON. Peristiwa masuk mewakili permintaan masuk tunggal dan mencakup informasi tentang penanggung jawab masuk, Wilayah, serta tanggal dan waktu tindakan.

Contoh peristiwa sukses masuk dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk acara login yang berhasil.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JohnDoe",
    "accountId": "111122223333",
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
```

```
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```

Untuk detail selengkapnya tentang informasi yang terkandung dalam file CloudTrail log, lihat Referensi CloudTrail Acara di AWS CloudTrail Panduan Pengguna.

Contoh peristiwa kegagalan masuk dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk peristiwa login gagal.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
```

```
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

Dari informasi ini, Anda dapat menentukan bahwa upaya masuk dilakukan oleh IAM pengguna bernama`JaneDoe`, seperti yang ditunjukkan dalam `userIdentity` elemen. Anda juga dapat melihat bahwa upaya masuk gagal, seperti yang ditunjukkan pada elemen `responseElements`. Anda dapat melihat bahwa `JaneDoe` mencoba masuk ke SNS konsol Amazon pada pukul 17:35 (UTC) pada 8 Juli 2014.

Contoh peristiwa gagal masuk yang disebabkan oleh nama pengguna yang salah

Contoh berikut menunjukkan entri CloudTrail log untuk peristiwa login yang gagal yang disebabkan oleh pengguna memasukkan nama pengguna yang salah. AWS menutupi `userName` teks dengan `HIDDEN_DUE_TO_SECURITY_REASONS` untuk membantu mencegah mengekspos informasi yang berpotensi sensitif.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
```

```
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
}
```

IAMperilaku kebijakan kepercayaan peran

Pada tanggal 21 September 2022, AWS membuat perubahan pada perilaku kebijakan kepercayaan IAM peran untuk meminta izin eksplisit dalam kebijakan kepercayaan peran ketika peran mengasumsikan dirinya sendiri. IAMperan dalam daftar izin perilaku lama memiliki additionalEventData bidang yang ada explicitTrustGrant untuk `AssumeRole` acara. Nilai false ketika peran pada daftar izin lama mengasumsikan dirinya menggunakan perilaku lama. `explicitTrustGrant` Ketika peran pada daftar izin lama mengasumsikan dirinya sendiri tetapi perilaku kebijakan kepercayaan peran telah diperbarui untuk secara eksplisit memungkinkan peran tersebut untuk mengambil alih dirinya sendiri, nilainya benar. `explicitTrustGrant`

Hanya sejumlah kecil IAM peran yang ada di daftar izinkan untuk perilaku lama, dan bidang ini hanya ada di CloudTrail log untuk peran ini ketika mereka mengambil peran tersebut. Dalam kebanyakan kasus, IAM peran tidak perlu diasumsikan sendiri. AWS merekomendasikan memperbarui proses, kode, atau konfigurasi Anda untuk menghapus perilaku ini atau memperbarui kebijakan kepercayaan peran Anda untuk secara eksplisit mengizinkan perilaku ini. Untuk informasi selengkapnya, lihat Mengumumkan pembaruan perilaku kebijakan kepercayaan IAM peran.

Validasi kepatuhan untuk AWS Identity and Access Management

Auditor pihak ketiga menilai keamanan dan kepatuhan AWS Identity and Access Management (IAM) sebagai bagian dari beberapa program AWS kepatuhan. Ini termasukSOC,PCI, Fed RAMPISO, dan lainnya.

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- Arsitektur untuk HIPAA Keamanan dan Kepatuhan di Amazon Web Services — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat HIPAA aplikasi yang memenuhi syarat.

  > **Note**
  >
  > Tidak semua Layanan AWS HIPAA memenuhi syarat. Untuk informasi selengkapnya, lihat Referensi Layanan yang HIPAA Memenuhi Syarat.

- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi ()). ISO
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas yang mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan

kepatuhan, seperti PCIDSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.

- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Ketahanan di AWS Identity and Access Management

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. AWS Wilayah memiliki beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat Infrastruktur AWS Global.

AWS Identity and Access Management (IAM) dan AWS Security Token Service (AWS STS) adalah layanan mandiri berbasis Wilayah yang tersedia secara global.

IAMadalah kritis Layanan AWS. Setiap operasi yang dilakukan AWS harus diautentikasi dan disahkan olehIAM. IAMmemeriksa setiap permintaan terhadap identitas dan kebijakan yang disimpan IAM untuk menentukan apakah permintaan diizinkan atau ditolak. IAMdirancang dengan bidang kontrol dan bidang data terpisah sehingga layanan mengautentikasi bahkan selama kegagalan yang tidak terduga. IAMsumber daya yang digunakan dalam otorisasi, seperti peran dan kebijakan, disimpan di bidang kontrol. IAMpelanggan dapat mengubah konfigurasi sumber daya ini dengan menggunakan IAM operasi seperti `DeletePolicy` dan`AttachRolePolicy`. Permintaan perubahan konfigurasi tersebut masuk ke bidang kontrol. Ada satu pesawat IAM kontrol untuk semua komersial Wilayah AWS, yang terletak di Wilayah AS Timur (Virginia N.). IAMSistem kemudian menyebarkan perubahan konfigurasi ke bidang IAM data di setiap diaktifkan Wilayah AWS. Bidang IAM data pada dasarnya adalah replika read-only dari data konfigurasi bidang IAM kontrol. Masing-masing Wilayah AWS memiliki instance yang sepenuhnya independen dari bidang IAM data, yang melakukan otentikasi dan otorisasi untuk permintaan dari Wilayah yang sama. Di setiap Wilayah, pesawat IAM data didistribusikan di setidaknya tiga Availability Zone, dan memiliki kapasitas yang cukup untuk mentolerir hilangnya Availability Zone tanpa gangguan pelanggan. Baik pesawat IAM kontrol dan data dibangun untuk nol waktu henti yang direncanakan, dengan semua pembaruan perangkat lunak dan operasi penskalaan dilakukan dengan cara yang tidak terlihat oleh pelanggan.

AWS STS permintaan selalu pergi ke satu titik akhir global secara default. Anda dapat menggunakan AWS STS endpoint Regional untuk mengurangi latensi atau memberikan redundansi tambahan untuk aplikasi Anda. Untuk mempelajari selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Peristiwa tertentu dapat mengganggu komunikasi antara Wilayah AWS melalui jaringan. Namun, bahkan ketika Anda tidak dapat berkomunikasi dengan IAM titik akhir global, masih AWS STS dapat mengautentikasi IAM prinsipal dan IAM dapat mengotorisasi permintaan Anda. Detail spesifik dari suatu peristiwa yang mengganggu komunikasi akan menentukan kemampuan Anda untuk mengakses AWS layanan. Dalam kebanyakan situasi, Anda dapat terus menggunakan IAM kredensil di lingkungan Anda AWS . Kondisi berikut mungkin berlaku untuk peristiwa yang mengganggu komunikasi.

Kunci akses untuk IAM pengguna

Anda dapat mengautentikasi tanpa batas waktu di Wilayah dengan kunci akses jangka panjang untuk IAM pengguna. Saat Anda menggunakan AWS Command Line Interface danAPIs, Anda dapat memberikan kunci AWS akses sehingga AWS dapat memverifikasi identitas Anda dalam permintaan terprogram.

> **Important**
>
> Sebagai praktik terbaik, kami menyarankan agar pengguna Anda masuk dengan kredensi sementara, bukan kunci akses jangka panjang.

Kredensial sementara

Anda dapat meminta kredensi sementara baru dengan titik akhir layanan AWS STS Regional setidaknya selama 24 jam. APIOperasi berikut menghasilkan kredensil sementara.

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAML
- GetFederationToken
- GetSessionToken

Prinsipal dan izin

- Anda mungkin tidak dapat menambahkan, memodifikasi, atau menghapus prinsip atau izin di. IAM
- Kredensi Anda mungkin tidak mencerminkan perubahan pada izin yang baru saja Anda terapkan. IAM Untuk informasi selengkapnya, lihat Perubahan yang saya buat tidak selalu langsung terlihat.

AWS Management Console

- Anda mungkin dapat menggunakan titik akhir masuk Regional untuk masuk ke AWS Management Console sebagai IAM pengguna. Titik akhir masuk regional memiliki format berikutURL.

  ```
  https://{Account ID}.signin.aws.amazon.com/console?region={Region}
  ```

  Contoh: https://111122223333.signin.aws.amazon.com /console? wilayah=us-barat-2
- Anda mungkin tidak dapat menyelesaikan otentikasi multi-faktor Universal 2nd Factor (U2F) (). MFA

Praktik terbaik untuk IAM ketahanan

AWS telah membangun ketahanan ke dalam Wilayah AWS dan Availability Zones. Ketika Anda mengamati praktik IAM terbaik berikut dalam sistem yang berinteraksi dengan lingkungan Anda, Anda memanfaatkan ketahanan itu.

1. Gunakan endpoint layanan AWS STS Regional, bukan endpoint global default.
2. Tinjau konfigurasi lingkungan Anda untuk sumber daya vital yang secara rutin membuat atau memodifikasi IAM sumber daya, dan menyiapkan solusi fallback yang menggunakan sumber daya yang ada. IAM

Keamanan infrastruktur dalam AWS Identity and Access Management

Sebagai layanan terkelola, AWS Identity and Access Management dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan API panggilan yang AWS dipublikasikan untuk mengakses IAM melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Transportasi (TLS). Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.

- Suite cipher dengan kerahasiaan maju yang sempurna (PFS) seperti (Ephemeral Diffie-Hellman) atau DHE (Elliptic Curve Ephemeral Diffie-Hellman). ECDHE Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani dengan menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan IAM prinsipal. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

IAMdapat diakses secara terprogram dengan menggunakan IAM HTTPSAPI, yang memungkinkan Anda mengeluarkan HTTPS permintaan langsung ke layanan. Query API mengembalikan informasi sensitif, termasuk kredensi keamanan. Karena itu, Anda harus menggunakan HTTPS dengan semua API permintaan. Saat Anda menggunakan HTTPSAPI, Anda harus menyertakan kode untuk menandatangani permintaan secara digital menggunakan kredensi Anda.

Anda dapat memanggil API operasi ini dari lokasi jaringan mana pun, IAM tetapi mendukung kebijakan akses berbasis sumber daya, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan IAM kebijakan untuk mengontrol akses dari titik akhir Amazon Virtual Private Cloud (AmazonVPC) tertentu atau spesifikVPCs. Secara efektif, ini mengisolasi akses jaringan ke IAM sumber daya tertentu hanya dari yang spesifik VPC dalam AWS jaringan.

Analisis konfigurasi dan kerentanan di AWS Identity and Access Management

AWS menangani tugas-tugas keamanan dasar seperti sistem operasi tamu (OS) dan patching database, konfigurasi firewall, dan pemulihan bencana. Prosedur ini telah ditinjau dan disertifikasi oleh pihak ketiga yang sesuai. Untuk detail selengkapnya, lihat sumber daya berikut:

- Model Tanggung Jawab Bersama
- Amazon Web Services: Gambaran Umum Proses Keamanan (whitepaper)

Sumber daya berikut juga membahas konfigurasi dan analisis kerentanan di AWS Identity and Access Management (IAM):

- Validasi kepatuhan untuk AWS Identity and Access Management
- Praktik terbaik keamanan dan kasus penggunaan AWS Identity and Access Management

AWS kebijakan terkelola untuk AWS Identity and Access Management Access Analyzer

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pemutakhiran akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau API operasi baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat kebijakan AWS terkelola di Panduan IAM Pengguna.

IAMReadOnlyAccess

Gunakan kebijakan `IAMReadOnlyAccess` terkelola untuk mengizinkan akses baca saja ke IAM sumber daya. Kebijakan ini memberikan izin untuk mendapatkan dan mencantumkan semua IAM sumber daya. Ini memungkinkan melihat detail dan laporan aktivitas untuk pengguna, grup, peran, kebijakan, penyedia identitas, dan MFA perangkat. Ini tidak termasuk kemampuan untuk membuat atau menghapus sumber daya atau akses ke sumber daya IAM Access Analyzer. Lihat kebijakan untuk daftar lengkap layanan dan tindakan yang didukung oleh kebijakan ini.

IAMUserChangePassword

Gunakan kebijakan `IAMUserChangePassword` terkelola untuk memungkinkan IAM pengguna mengubah kata sandi mereka.

Anda mengonfigurasi pengaturan IAM Akun dan kebijakan Kata Sandi untuk memungkinkan IAM pengguna mengubah kata sandi IAM akun mereka. Saat Anda mengizinkan tindakan ini, IAM lampirkan kebijakan berikut ke setiap pengguna:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

IAMAccessAnalyzerFullAccess

Gunakan kebijakan `IAMAccessAnalyzerFullAccess` AWS terkelola untuk mengizinkan administrator mengakses IAM Access Analyzer.

Pengelompokan izin

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- IAMAccess Analyzer - Memungkinkan izin administratif penuh ke semua sumber daya di IAM Access Analyzer.
- Buat peran terkait layanan — Memungkinkan administrator membuat peran terkait layanan, yang memungkinkan IAM Access Analyzer menganalisis sumber daya di layanan lain atas nama Anda. Izin ini memungkinkan pembuatan peran terkait layanan hanya untuk digunakan oleh IAM Access Analyzer.

- AWS Organizations— Memungkinkan administrator untuk menggunakan IAM Access Analyzer untuk organisasi di. AWS Organizations Setelah mengaktifkan akses tepercaya untuk IAM Access Analyzer di AWS Organizations, anggota akun manajemen dapat melihat temuan di seluruh organisasi mereka.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
```

```
  ]
}
```

IAMAccessAnalyzerReadOnlyAccess

Gunakan kebijakan `IAMAccessAnalyzerReadOnlyAccess` AWS terkelola untuk mengizinkan akses hanya-baca ke IAM Access Analyzer.

Untuk juga mengizinkan akses hanya-baca ke IAM Access Analyzer AWS Organizations, buat kebijakan terkelola pelanggan yang memungkinkan tindakan Deskripsikan dan Daftar dari kebijakan IAMAccessAnalyzerFullAccess AWS terkelola.

Izin tingkat layanan

Kebijakan ini menyediakan akses hanya-baca ke IAM Access Analyzer. Tidak ada izin layanan lain yang disertakan dalam kebijakan ini.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "access-analyzer:CheckAccessNotGranted",
        "access-analyzer:CheckNoNewAccess",
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRolePolicy

Anda tidak dapat melampirkan AccessAnalyzerServiceRolePolicy ke IAM entitas Anda. Kebijakan ini dilampirkan pada peran terkait layanan yang memungkinkan IAM Access Analyzer melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer.

Pengelompokan izin

Kebijakan ini memungkinkan akses ke IAM Access Analyzer untuk menganalisis metadata sumber daya dari beberapa. Layanan AWS

- Amazon DynamoDB - Memungkinkan izin untuk melihat aliran dan tabel DynamoDB.
- Amazon Elastic Compute Cloud — Memungkinkan izin untuk mendeskripsikan alamat IP, snapshot, dan. VPCs
- Amazon Elastic Container Registry - Memungkinkan izin untuk mendeskripsikan repositori gambar dan mengambil kebijakan repositori.
- Amazon Elastic File System - Memungkinkan izin untuk melihat deskripsi sistem EFS file Amazon dan melihat kebijakan tingkat sumber daya untuk sistem file Amazon. EFS
- AWS Identity and Access Management— Memungkinkan izin untuk mengambil informasi tentang peran tertentu dan daftar IAM peran yang memiliki awalan jalur tertentu. Memungkinkan izin untuk mengambil informasi tentang pengguna, IAM grup, profil login, kunci akses, dan layanan data yang terakhir diakses.
- AWS Key Management Service— Memungkinkan izin untuk melihat informasi terperinci tentang KMS kunci dan kebijakan dan hibah utamanya.
- AWS Lambda— Memungkinkan izin untuk melihat informasi tentang alias, fungsi, lapisan, dan alias Lambda.
- AWS Organizations— Memungkinkan izin untuk Organizations dan memungkinkan pembuatan analyzer dalam AWS organisasi sebagai zona kepercayaan.
- Amazon Relational Database Service - Memungkinkan izin untuk melihat informasi terperinci tentang snapshot RDS Amazon DB dan snapshot cluster RDS Amazon DB.
- Amazon Simple Storage Service - Memungkinkan izin untuk melihat informasi terperinci tentang jalur akses Amazon S3, bucket, dan bucket direktori Amazon S3 yang menggunakan kelas penyimpanan Amazon S3 Express One.
- AWS Secrets Manager— Memungkinkan izin untuk melihat informasi rinci tentang rahasia dan kebijakan sumber daya yang melekat pada rahasia.
- Amazon Simple Notification Service - Memungkinkan izin untuk melihat informasi rinci tentang suatu topik.
- Amazon Simple Queue Service - Memungkinkan izin untuk melihat informasi rinci tentang antrian tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessAnalyzerServiceRolePolicy",
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetResourcePolicy",
        "dynamodb:ListStreams",
        "dynamodb:ListTables",
        "ec2:DescribeAddresses",
        "ec2:DescribeByoipCidrs",
        "ec2:DescribeSnapshotAttribute",
        "ec2:DescribeSnapshots",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcs",
        "ec2:GetSnapshotBlockPublicAccessState",
        "ecr:DescribeRepositories",
        "ecr:GetRepositoryPolicy",
        "elasticfilesystem:DescribeFileSystemPolicy",
        "elasticfilesystem:DescribeFileSystems",
        "iam:GetRole",
        "iam:ListEntitiesForPolicy",
        "iam:ListRoles",
        "iam:ListUsers",
        "iam:GetUser",
        "iam:GetGroup",
        "iam:GenerateServiceLastAccessedDetails",
        "iam:GetServiceLastAccessedDetails",
        "iam:ListAccessKeys",
        "iam:GetLoginProfile",
        "iam:GetAccessKeyLastUsed",
        "iam:ListRolePolicies",
        "iam:GetRolePolicy",
        "iam:ListAttachedRolePolicies",
        "iam:ListUserPolicies",
        "iam:GetUserPolicy",
        "iam:ListAttachedUserPolicies",
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListGroupsForUser",
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
```

```
        "kms:ListGrants",
        "kms:ListKeyPolicies",
        "kms:ListKeys",
        "lambda:GetFunctionUrlConfig",
        "lambda:GetLayerVersionPolicy",
        "lambda:GetPolicy",
        "lambda:ListAliases",
        "lambda:ListFunctions",
        "lambda:ListLayers",
        "lambda:ListLayerVersions",
        "lambda:ListVersionsByFunction",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots",
        "rds:DescribeDBClusterSnapshotAttributes",
        "rds:DescribeDBClusterSnapshots",
        "rds:DescribeDBSnapshotAttributes",
        "rds:DescribeDBSnapshots",
        "s3:DescribeMultiRegionAccessPointOperation",
        "s3:GetAccessPoint",
        "s3:GetAccessPointPolicy",
        "s3:GetAccessPointPolicyStatus",
        "s3:GetAccountPublicAccessBlock",
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicyStatus",
        "s3:GetBucketPolicy",
        "s3:GetBucketPublicAccessBlock",
        "s3:GetMultiRegionAccessPoint",
        "s3:GetMultiRegionAccessPointPolicy",
        "s3:GetMultiRegionAccessPointPolicyStatus",
        "s3:ListAccessPoints",
        "s3:ListAllMyBuckets",
        "s3:ListMultiRegionAccessPoints",
        "s3express:GetBucketPolicy",
        "s3express:ListAllMyDirectoryBuckets",
```

```
        "sns:GetTopicAttributes",
        "sns:ListTopics",
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:ListSecrets",
        "sqs:GetQueueAttributes",
        "sqs:ListQueues"
      ],
      "Resource": "*"
    }
  ]
}
```

IAMdan pembaruan IAM Access Analyzer ke kebijakan AWS terkelola

Lihat detail tentang pembaruan IAM dan kebijakan AWS terkelola sejak layanan mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan RSS umpan di halaman riwayat Dokumen Penganalisis IAM Akses IAM dan Akses.

Perubahan	Deskripsi	Tanggal
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang kebijakan IAM pengguna dan peran ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	30 Mei 2024
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan dukungan untuk izin untuk mengambil status saat ini dari blokir akses publik untuk EC2 snapshot Amazon ke izin	23 Januari 2024

Perubahan	Deskripsi	Tanggal
	tingkat layanan. `AccessAnalyzerServiceRolePolicy`	
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan dukungan untuk aliran dan tabel DynamoDB ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	Januari 11, 2024
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan dukungan untuk bucket direktori Amazon S3 ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	1 Desember 2023
IAMAccessAnalyzerReadOnlyAccess— Izin Ditambahkan	IAMAccess Analyzer menambahkan izin untuk memungkinkan Anda memeriksa apakah pembaruan kebijakan Anda memberikan akses tambahan. Izin ini diperlukan oleh IAM Access Analyzer untuk melakukan pemeriksaan kebijakan pada kebijakan Anda.	26 November 2023

Perubahan	Deskripsi	Tanggal
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan IAM tindakan ke izin tingkat layanan `AccessAnalyzerServiceRolePolicy` untuk mendukung tindakan berikut: • Daftar entitas untuk kebijakan • Menghasilkan detail layanan yang terakhir diakses • Daftar informasi kunci akses	26 November 2023
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan dukungan untuk jenis sumber daya berikut ke izin tingkat layanan: `AccessAnalyzerServiceRolePolicy` • Cuplikan EBS volume Amazon • ECRRepositori Amazon • Sistem EFS file Amazon • Cuplikan Amazon RDS DB • Cuplikan kluster Amazon RDS DB • SNSTopik Amazon	25 Oktober 2022

Perubahan	Deskripsi	Tanggal
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan `lambda:GetFunctionUrlConfig` tindakan ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	April 6, 2022
AccessAnalyzerServiceRolePolicy— Izin Ditambahkan	IAMAccess Analyzer menambahkan tindakan Amazon S3 baru untuk menganalisis metadata yang terkait dengan titik akses multi-wilayah.	2 September 2021
IAMAccessAnalyzerReadOnlyAccess— Izin Ditambahkan	IAMAccess Analyzer menambahkan tindakan baru untuk memberikan `ValidatePolicy` izin agar Anda dapat menggunakan pemeriksaan kebijakan untuk validasi. Izin ini diperlukan oleh IAM Access Analyzer untuk melakukan pemeriksaan kebijakan pada kebijakan Anda.	16 Maret 2021
IAMAccess Analyzer mulai melacak perubahan	IAMAccess Analyzer mulai melacak perubahan untuk kebijakan yang AWS dikelola.	1 Maret 2021

Fitur keamanan di luar IAM

Anda menggunakan IAM untuk mengontrol akses ke tugas yang dilakukan menggunakan AWS Management Console, Alat Baris AWS Perintah, atau operasi API layanan menggunakan AWS SDK. Beberapa AWS produk memiliki cara lain untuk mengamankan sumber daya mereka juga. Daftar berikut ini memberikan beberapa contoh, meskipun tidak lengkap.

Amazon EC2

Di Amazon Elastic Compute Cloud, Anda masuk ke suatu instans dengan key pair (untuk instans Linux) atau menggunakan nama pengguna dan kata sandi (untuk instans Microsoft Windows).

Untuk informasi selengkapnya, lihat dokumentasi berikut ini:

- Memulai Instans Linux Amazon EC2 di Panduan Pengguna Amazon EC2
- Memulai Instans Windows Amazon EC2 di Panduan Pengguna Amazon EC2

Amazon RDS

Di Amazon Relational Database Service, Anda masuk ke mesin basis data dengan nama pengguna dan kata sandi yang terikat dengan basis data tersebut.

Untuk informasi selengkapnya, lihat Memulai dengan Amazon RDS di Panduan Pengguna Amazon RDS.

Amazon EC2 dan Amazon RDS

Di Amazon EC2 dan Amazon RDS, Anda menggunakan grup keamanan untuk mengontrol lalu lintas ke suatu instans atau basis data.

Untuk informasi selengkapnya, lihat dokumentasi berikut ini:

- Grup Keamanan Amazon EC2 untuk Instans Linux di Panduan Pengguna Amazon EC2
- Grup Keamanan Amazon EC2 untuk Instans Windows di Panduan Pengguna Amazon EC2
- Grup Keamanan Amazon RDS di Panduan Pengguna Amazon RDS

WorkSpaces

Di Amazon WorkSpaces, pengguna masuk ke desktop dengan nama pengguna dan kata sandi.

Untuk informasi selengkapnya, lihat Memulai WorkSpaces di Panduan WorkSpaces Administrasi Amazon.

Amazon WorkDocs

Di Amazon WorkDocs, pengguna mendapatkan akses ke dokumen bersama dengan masuk dengan nama pengguna dan kata sandi.

Untuk informasi selengkapnya, lihat Memulai Amazon WorkDocs di Panduan WorkDocs Administrasi Amazon.

Metode kontrol akses ini bukan bagian dari IAM. IAM memungkinkan Anda mengontrol cara AWS produk ini dikelola — membuat atau menghentikan instans Amazon EC2, menyiapkan desktop baru, dan sebagainya. WorkSpaces Artinya, IAM membantu Anda mengontrol tugas yang dilakukan dengan membuat permintaan ke Amazon Web Services, dan ini membantu Anda mengontrol akses ke AWS Management Console. Namun, IAM tidak membantu Anda mengelola keamanan untuk tugas-tugas seperti masuk ke sistem operasi (Amazon EC2), database (Amazon RDS), desktop (Amazon), atau situs kolaborasi (WorkSpacesAmazon). WorkDocs

Saat Anda bekerja dengan AWS produk tertentu, pastikan untuk membaca dokumentasi untuk mempelajari opsi keamanan untuk semua sumber daya milik produk tersebut.

Menggunakan AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer menyediakan kemampuan berikut:

- IAMAccess Analyzer External Access Analyzer membantu mengidentifikasi sumber daya di organisasi dan akun Anda yang dibagikan dengan entitas eksternal.
- IAMAccess Analyzer penganalisis akses yang tidak digunakan membantu mengidentifikasi akses yang tidak digunakan di organisasi dan akun Anda.
- IAMAccess Analyzer memvalidasi IAM kebijakan terhadap tata bahasa kebijakan dan AWS praktik terbaik.
- IAMPemeriksaan kebijakan kustom Access Analyzer membantu memvalidasi IAM kebijakan terhadap standar keamanan yang Anda tentukan.
- IAMAccess Analyzer menghasilkan IAM kebijakan berdasarkan aktivitas akses di AWS CloudTrail log Anda.

Mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal

IAMAccess Analyzer membantu Anda mengidentifikasi sumber daya di organisasi dan akun Anda, seperti bucket IAM atau peran Amazon S3, yang dibagikan dengan entitas eksternal. Hal ini memungkinkan Anda mengidentifikasi akses yang tidak diinginkan ke sumber daya dan data Anda, yang merupakan sebuah risiko keamanan. IAMAccess Analyzer mengidentifikasi sumber daya yang dibagikan dengan prinsipal eksternal dengan menggunakan penalaran berbasis logika untuk menganalisis kebijakan berbasis sumber daya di lingkungan Anda. AWS Untuk setiap instance sumber daya yang dibagikan di luar akun Anda, IAM Access Analyzer menghasilkan temuan. Temuan mencakup informasi tentang akses dan prinsip eksternal yang diberikan kepadanya. Anda dapat meninjau temuan untuk menentukan apakah akses dimaksudkan dan aman atau jika akses tidak diinginkan dan risiko keamanan. Selain membantu mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal, Anda dapat menggunakan temuan IAM Access Analyzer untuk melihat bagaimana kebijakan Anda memengaruhi akses publik dan lintas akun ke sumber daya Anda sebelum menerapkan izin sumber daya. Temuan ini diatur dalam dasbor ringkasan visual. Dasbor menyoroti pemisahan antara temuan akses publik dan lintas akun, dan memberikan rincian temuan

berdasarkan jenis sumber daya. Untuk mempelajari lebih lanjut tentang dasbor, lihatLihat dasbor temuan IAM Access Analyzer.

> **Note**
>
> Entitas eksternal dapat berupa AWS akun lain, pengguna root, IAM pengguna atau peran, pengguna federasi, pengguna anonim, atau entitas lain yang dapat Anda gunakan untuk membuat filter. Untuk informasi selengkapnya, lihat Elemen AWS JSON Kebijakan: Principal.

Saat mengaktifkan IAM Access Analyzer, Anda membuat analisa untuk seluruh organisasi atau akun Anda. Akun atau organisasi yang Anda pilih dikenal sebagai zona kepercayaan untuk penganalisis. Penganalisis memantau semua sumber daya yang didukung dalam zona kepercayaan Anda. Setiap akses ke sumber daya oleh kepala sekolah dalam zona kepercayaan Anda dianggap tepercaya. Setelah diaktifkan, IAM Access Analyzer menganalisis kebijakan yang diterapkan ke semua sumber daya yang didukung di zona kepercayaan Anda. Setelah analisis pertama, IAM Access Analyzer menganalisis kebijakan ini secara berkala. Jika Anda menambahkan kebijakan baru, atau mengubah kebijakan yang ada, IAM Access Analyzer akan menganalisis kebijakan baru atau yang diperbarui dalam waktu sekitar 30 menit.

Saat menganalisis kebijakan, jika IAM Access Analyzer mengidentifikasi kebijakan yang memberikan akses ke prinsipal eksternal yang tidak berada dalam zona kepercayaan Anda, itu menghasilkan temuan. Setiap temuan mencakup rincian tentang sumber daya, entitas eksternal dengan akses ke sana, dan izin yang diberikan sehingga Anda dapat mengambil tindakan yang tepat. Anda dapat melihat detail yang disertakan dalam temuan untuk menentukan apakah akses sumber daya disengaja atau merupakan potensi risiko yang harus Anda selesaikan. Saat menambahkan kebijakan ke sumber daya, atau memperbarui kebijakan yang ada, IAM Access Analyzer akan menganalisis kebijakan tersebut. IAMAccess Analyzer juga menganalisis semua kebijakan berbasis sumber daya secara berkala.

Pada kesempatan langka dalam kondisi tertentu, IAM Access Analyzer tidak menerima pemberitahuan tentang kebijakan yang ditambahkan atau diperbarui, yang dapat menyebabkan keterlambatan dalam temuan yang dihasilkan. IAMAccess Analyzer dapat memakan waktu hingga 6 jam untuk menghasilkan atau menyelesaikan temuan jika Anda membuat atau menghapus titik akses multi-wilayah yang terkait dengan bucket Amazon S3, atau memperbarui kebijakan untuk jalur akses multi-wilayah. Selain itu, jika ada masalah pengiriman dengan pengiriman AWS CloudTrail log, perubahan kebijakan tidak memicu pemindaian ulang sumber daya yang dilaporkan dalam temuan tersebut. Ketika ini terjadi, IAM Access Analyzer menganalisis kebijakan

baru atau yang diperbarui selama pemindaian berkala berikutnya, yaitu dalam 24 jam. Jika Anda ingin mengonfirmasi perubahan yang Anda buat pada kebijakan menyelesaikan masalah akses yang dilaporkan dalam temuan, Anda dapat memindai ulang sumber daya yang dilaporkan dalam temuan menggunakan tautan Pindai Ulang di halaman detail Temuan, atau dengan menggunakan `StartResourceScan`pengoperasian Access Analyzer. IAM API Untuk mempelajari selengkapnya, lihat Selesaikan IAM temuan Access Analyzer.

> **Important**
>
> IAMAccess Analyzer hanya menganalisis kebijakan yang diterapkan pada sumber daya di AWS Wilayah yang sama dengan yang diaktifkan. Untuk memantau semua sumber daya di AWS lingkungan Anda, Anda harus membuat penganalisis untuk mengaktifkan IAM Access Analyzer di setiap Wilayah tempat Anda menggunakan sumber daya yang didukung AWS .

IAMAccess Analyzer menganalisis jenis sumber daya berikut:

- Bucket Amazon Simple Storage Service
- Ember direktori Layanan Penyimpanan Sederhana Amazon
- AWS Identity and Access Management peran
- AWS Key Management Service kunci
- AWS Lambda fungsi dan lapisan
- Antrean Amazon Simple Queue Service
- AWS Secrets Manager rahasia
- Topik Amazon Simple Notification Service
- Cuplikan volume Amazon Elastic Block Store
- Cuplikan DB Layanan Amazon Relational Database Service
- Cuplikan cluster DB Layanan Relational Database Service Amazon
- Repositori Registri Wadah Elastis Amazon
- Sistem file Amazon Elastic File System
- Aliran Amazon DynamoDB
- Tabel Amazon DynamoDB

Mengidentifikasi akses yang tidak terpakai yang diberikan kepada IAM pengguna dan peran

IAMAccess Analyzer membantu Anda mengidentifikasi dan meninjau akses yang tidak digunakan di AWS organisasi dan akun Anda. IAMAccess Analyzer terus memantau semua IAM peran dan pengguna di AWS organisasi dan akun Anda dan menghasilkan temuan untuk akses yang tidak digunakan. Temuan ini menyoroti peran yang tidak digunakan, kunci akses yang tidak digunakan untuk IAM pengguna, dan kata sandi yang tidak digunakan untuk pengguna. IAM Untuk IAM peran aktif dan pengguna, temuan ini memberikan visibilitas ke layanan dan tindakan yang tidak digunakan.

Temuan untuk akses eksternal dan penganalisis akses yang tidak digunakan diatur ke dalam dasbor ringkasan visual. Dasbor menyoroti Anda Akun AWS yang memiliki temuan paling banyak dan memberikan rincian temuan berdasarkan jenisnya. Untuk informasi selengkapnya tentang dasbor, lihat Lihat dasbor temuan IAM Access Analyzer.

IAMAccess Analyzer meninjau informasi yang terakhir diakses untuk semua peran di AWS organisasi dan akun Anda untuk membantu Anda mengidentifikasi akses yang tidak digunakan. IAMtindakan informasi yang diakses terakhir membantu Anda mengidentifikasi tindakan yang tidak digunakan untuk peran dalam Anda Akun AWS. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Memvalidasi kebijakan terhadap praktik AWS terbaik

Anda dapat memvalidasi kebijakan Anda terhadap tata bahasa IAM kebijakan dan praktik AWS terbaik menggunakan pemeriksaan kebijakan dasar yang disediakan oleh validasi kebijakan IAM Access Analyzer. Anda dapat membuat atau mengedit kebijakan menggunakan AWS CLI, AWS API, atau editor JSON kebijakan di IAM konsol. Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan Anda. Temuan ini memberikan rekomendasi yang dapat ditindaklanjuti yang membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik. AWS Untuk mempelajari lebih lanjut tentang memvalidasi kebijakan menggunakan validasi kebijakan, lihat. Validasi kebijakan dengan IAM Access Analyzer

Memvalidasi kebijakan terhadap standar keamanan yang Anda tentukan

Anda dapat memvalidasi kebijakan Anda terhadap standar keamanan yang ditentukan menggunakan pemeriksaan kebijakan kustom IAM Access Analyzer. Anda dapat membuat atau mengedit kebijakan menggunakan AWS CLI, AWS API, atau editor JSON kebijakan di IAM konsol. Melalui konsol, Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan versi yang ada. Melalui AWS CLI dan AWS API, Anda juga dapat memeriksa IAM tindakan spesifik yang Anda anggap penting tidak diizinkan oleh kebijakan. Pemeriksaan ini menyoroti pernyataan kebijakan yang memberikan akses baru. Anda dapat memperbarui pernyataan kebijakan dan menjalankan kembali pemeriksaan hingga kebijakan sesuai dengan standar keamanan Anda. Untuk mempelajari selengkapnya tentang memvalidasi kebijakan menggunakan pemeriksaan kebijakan khusus, lihatValidasi kebijakan dengan pemeriksaan kebijakan khusus IAM Access Analyzer.

Membuat kebijakan

IAMAccess Analyzer menganalisis AWS CloudTrail log Anda untuk mengidentifikasi tindakan dan layanan yang telah digunakan oleh IAM entitas (pengguna atau peran) dalam rentang tanggal yang ditentukan. Kemudian menghasilkan IAM kebijakan yang didasarkan pada aktivitas akses tersebut. Anda dapat menggunakan kebijakan yang dihasilkan untuk menyaring izin entitas dengan melampirkannya ke IAM pengguna atau peran. Untuk mempelajari selengkapnya tentang membuat kebijakan menggunakan IAM Access Analyzer, lihatIAMPembuatan kebijakan Access Analyzer.

Harga untuk IAM Access Analyzer

IAMAccess Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per penganalisis per bulan.

- Anda akan dikenakan biaya untuk setiap penganalisis akses yang tidak terpakai yang Anda buat.
- Membuat penganalisis akses yang tidak digunakan di beberapa Wilayah akan mengakibatkan Anda dikenakan biaya untuk setiap penganalisis.
- Peran terkait layanan tidak dianalisis untuk aktivitas akses yang tidak digunakan dan peran tersebut tidak termasuk dalam jumlah total peran yang dianalisis. IAM

IAMAccess Analyzer mengenakan biaya untuk pemeriksaan kebijakan khusus berdasarkan jumlah API permintaan yang dibuat ke IAM Access Analyzer untuk memeriksa akses baru.

Untuk daftar lengkap biaya dan harga IAM Access Analyzer, lihat harga IAMAccess Analyzer.

Untuk melihat tagihan Anda, buka Dasbor Manajemen Penagihan dan Biaya di konsol AWS Billing and Cost Management. Tagihan Anda berisi tautan ke laporan penggunaan yang memberikan detail tentang tagihan Anda. Untuk mempelajari selengkapnya tentang Akun AWS penagihan, lihat AWS Billing Panduan Pengguna

Jika Anda memiliki pertanyaan tentang AWS penagihan, akun, dan acara, hubungi AWS Support.

Temuan untuk akses eksternal dan tidak terpakai

IAMAccess Analyzer menghasilkan temuan untuk akses eksternal dan akses yang tidak terpakai di organisasi Akun AWS atau Anda. Untuk akses eksternal, IAM Access Analyzer menghasilkan temuan untuk setiap instance kebijakan berbasis sumber daya yang memberikan akses ke sumber daya dalam zona kepercayaan Anda kepada prinsipal yang tidak berada dalam zona kepercayaan Anda. Saat Anda membuat penganalisis akses eksternal, Anda memilih organisasi atau Akun AWS menganalisis. Setiap penanggung jawab dalam organisasi atau akun yang Anda pilih untuk penganalisis dianggap tepercaya. Karena penanggung jawab dalam organisasi atau akun yang sama dipercaya, sumber daya dan penanggung jawab di dalam organisasi atau akun yang terdiri dari zona kepercayaan untuk penganalisis. Setiap berbagi yang berada dalam zona kepercayaan dianggap aman, sehingga IAM Access Analyzer tidak menghasilkan temuan. Misalnya, jika Anda memilih sebuah organisasi sebagai zona kepercayaan untuk seorang penganalisis, semua sumber daya dan penanggung jawab di organisasi berada di dalam zona kepercayaan. Jika Anda memberikan izin ke bucket Amazon S3 di salah satu akun anggota organisasi Anda ke kepala sekolah di akun anggota organisasi lainIAM, Access Analyzer tidak akan menghasilkan temuan. Tetapi jika Anda memberikan izin kepada kepala sekolah di akun yang bukan anggota organisasi, IAM Access Analyzer menghasilkan temuan.

IAMAccess Analyzer juga menghasilkan temuan untuk akses yang tidak terpakai yang diberikan di AWS organisasi dan akun Anda. Saat Anda membuat penganalisis akses yang tidak digunakan, IAM Access Analyzer terus memantau semua IAM peran dan pengguna di AWS organisasi dan akun Anda dan menghasilkan temuan untuk akses yang tidak digunakan. IAMAccess Analyzer menghasilkan jenis temuan berikut untuk akses yang tidak digunakan:

- Peran yang tidak digunakan — Peran tanpa aktivitas akses dalam jendela penggunaan yang ditentukan.
- Kunci akses IAM pengguna dan kata sandi yang tidak digunakan — Kredensi milik IAM pengguna yang belum digunakan untuk mengakses Anda Akun AWS di jendela penggunaan yang ditentukan.

- Izin yang tidak digunakan — Izin tingkat layanan dan tingkat tindakan yang tidak digunakan oleh peran dalam jendela penggunaan yang ditentukan. IAMAccess Analyzer menggunakan kebijakan berbasis identitas yang dilampirkan pada peran untuk menentukan layanan dan tindakan yang dapat diakses oleh peran tersebut. IAMAccess Analyzer mendukung peninjauan izin yang tidak digunakan untuk semua izin tingkat layanan. Untuk daftar lengkap izin tingkat tindakan yang didukung untuk temuan akses yang tidak digunakan, lihat. IAMtindakan terakhir diakses layanan informasi dan tindakan

> **Note**
>
> IAMAccess Analyzer menawarkan temuan akses eksternal secara gratis dan biaya untuk temuan akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per penganalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Topik

- Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer

Memahami cara kerja temuan IAM Access Analyzer

Topik ini menjelaskan konsep dan istilah yang digunakan dalam IAM Access Analyzer untuk membantu Anda menjadi akrab dengan cara IAM Access Analyzer memantau akses ke sumber daya Anda AWS .

Temuan akses eksternal

Temuan akses eksternal dihasilkan hanya sekali untuk setiap contoh sumber daya yang dibagikan di luar zona kepercayaan Anda. Setiap kali kebijakan berbasis sumber daya diubah, IAM Access Analyzer menganalisis kebijakan tersebut. Jika kebijakan yang diperbarui membagikan sumber daya yang sudah diidentifikasi dalam temuan, tetapi menggunakan izin atau ketentuan yang berbeda, temuan baru akan dihasilkan untuk contoh pembagian sumber daya tersebut. Jika akses dalam temuan pertama dihapus, temuan itu diperbarui ke status Terselesaikan.

Status semua temuan tetap aktif sampai Anda mengarsipkannya atau menghapus akses yang menghasilkan temuan. Saat Anda menghapus akses, status temuan diperbarui ke Terselesaikan.

> **Note**
>
> Diperlukan waktu hingga 30 menit setelah kebijakan diubah untuk IAM Access Analyzer untuk menganalisis sumber daya dan kemudian memperbarui temuan akses eksternal.

Bagaimana IAM Access Analyzer menghasilkan temuan untuk akses eksternal

AWS Identity and Access Management Access Analyzer Menggunakan teknologi yang disebut Zelkova untuk menganalisis IAM kebijakan dan mengidentifikasi akses eksternal ke sumber daya.

Zelkova menerjemahkan IAM kebijakan ke dalam pernyataan logis yang setara dan menjalankannya melalui serangkaian pemecah logis tujuan umum dan khusus (teori modulo kepuasan). IAMAccess Analyzer menerapkan Zelkova berulang kali pada kebijakan, menggunakan kueri yang semakin spesifik untuk mengkarakterisasi jenis akses yang diizinkan kebijakan berdasarkan kontennya. Untuk informasi lebih lanjut tentang teori modulo kepuasan, lihat Teori Modulo Kepuasan.

Untuk penganalisis akses eksternal, IAM Access Analyzer tidak memeriksa log akses untuk menentukan apakah entitas eksternal telah benar-benar mengakses sumber daya dalam zona

kepercayaan Anda. Sebaliknya, ini menghasilkan temuan ketika kebijakan berbasis sumber daya memungkinkan akses ke sumber daya, terlepas dari apakah sumber daya diakses oleh entitas eksternal.

Selain itu, IAM Access Analyzer tidak mempertimbangkan status akun eksternal apa pun saat membuat penentuannya. Jika ini menunjukkan bahwa akun 111122223333 dapat mengakses bucket Amazon S3 Anda, akun tersebut tidak memiliki informasi apa pun tentang pengguna, peran, kebijakan kontrol layanan (SCP), atau konfigurasi relevan lainnya di akun tersebut. Ini untuk privasi pelanggan, karena IAM Access Analyzer tidak tahu siapa yang memiliki akun lain. Ini juga untuk keamanan, karena penting untuk mengetahui tentang potensi akses eksternal bahkan jika saat ini tidak ada prinsip aktif yang dapat menggunakannya.

IAMAccess Analyzer hanya mempertimbangkan kunci IAM kondisi tertentu yang tidak dapat dipengaruhi secara langsung oleh pengguna eksternal atau yang berdampak pada otorisasi. Untuk contoh kunci kondisi yang dipertimbangkan IAM Access Analyzer, lihat Kunci filter IAM Access Analyzer.

IAMAccess Analyzer saat ini tidak melaporkan temuan dari Layanan AWS kepala sekolah atau akun layanan internal. Dalam kasus yang jarang terjadi di mana tidak dapat sepenuhnya menentukan apakah pernyataan kebijakan memberikan akses ke entitas eksternal, itu salah di sisi mendeklarasikan temuan positif palsu. Ini karena IAM Access Analyzer dirancang untuk memberikan pandangan komprehensif tentang berbagi sumber daya di akun Anda dan untuk meminimalkan negatif palsu.

Temuan akses yang tidak digunakan

Temuan akses yang tidak digunakan dihasilkan untuk IAM entitas dalam akun atau organisasi yang dipilih berdasarkan jumlah hari yang ditentukan saat membuat penganalisis. Temuan baru dihasilkan saat penganalisis memindai entitas berikutnya jika salah satu dari kondisi berikut terpenuhi:

- Peran tidak aktif untuk jumlah hari yang ditentukan.
- Izin yang tidak digunakan, kata sandi pengguna yang tidak digunakan, atau kunci akses pengguna yang tidak digunakan melampaui jumlah hari yang ditentukan.

> **Note**
>
> Temuan akses yang tidak digunakan hanya tersedia menggunakan API tindakan ListFindingsV2.

Bagaimana IAM Access Analyzer menghasilkan temuan untuk akses yang tidak digunakan

Untuk menganalisis akses yang tidak digunakan, Anda harus membuat penganalisis terpisah untuk temuan akses yang tidak digunakan untuk peran Anda, bahkan jika Anda telah membuat penganalisis untuk menghasilkan temuan akses eksternal untuk sumber daya Anda.

Setelah membuat penganalisis akses yang tidak digunakan, IAM Access Analyzer meninjau aktivitas akses untuk mengidentifikasi akses yang tidak digunakan. IAMAccess Analyzer memeriksa informasi terakhir yang diakses untuk semua peran, kunci akses pengguna, dan kata sandi pengguna di seluruh AWS organisasi dan akun Anda. Ini membantu Anda mengidentifikasi akses yang tidak digunakan.

Untuk IAM peran aktif dan pengguna, IAM Access Analyzer menggunakan informasi yang terakhir diakses untuk IAM layanan dan tindakan untuk mengidentifikasi izin yang tidak digunakan. Ini memungkinkan Anda untuk menskalakan proses peninjauan Anda di tingkat AWS organisasi dan akun. Anda juga dapat menggunakan tindakan informasi yang diakses terakhir untuk penyelidikan lebih dalam tentang peran individu. Ini memberikan wawasan yang lebih terperinci tentang izin tertentu yang tidak digunakan.

Dengan membuat penganalisis yang didedikasikan untuk akses yang tidak terpakai, Anda dapat meninjau dan mengidentifikasi akses yang tidak digunakan secara komprehensif di seluruh AWS lingkungan Anda, melengkapi temuan yang dihasilkan oleh penganalisis akses eksternal yang ada.

Memulai dengan AWS Identity and Access Management Access Analyzer temuan

Gunakan informasi dalam topik ini untuk mempelajari tentang persyaratan yang diperlukan untuk menggunakan dan mengelola AWS Identity and Access Management Access Analyzer, dan kemudian cara mengaktifkan IAM Access Analyzer. Untuk mempelajari lebih lanjut tentang peran terkait layanan untuk IAM Access Analyzer, lihat. Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer

Izin diperlukan untuk menggunakan IAM Access Analyzer

Agar berhasil mengkonfigurasi dan menggunakan IAM Access Analyzer, akun yang Anda gunakan harus diberikan izin yang diperlukan.

AWS kebijakan terkelola untuk IAM Access Analyzer

AWS Identity and Access Management Access Analyzer menyediakan kebijakan AWS terkelola untuk membantu Anda memulai dengan cepat.

- IAMAccessAnalyzerFullAccess- Memungkinkan akses penuh ke IAM Access Analyzer untuk administrator. Kebijakan ini juga memungkinkan pembuatan peran terkait layanan yang diperlukan untuk memungkinkan IAM Access Analyzer menganalisis sumber daya di akun atau organisasi Anda. AWS
- IAMAccessAnalyzerReadOnlyAccess- Memungkinkan akses read-only ke IAM Access Analyzer. Anda harus menambahkan kebijakan tambahan ke IAM identitas Anda (pengguna, grup pengguna, atau peran) agar mereka dapat melihat temuan mereka.

Sumber daya yang ditentukan oleh IAM Access Analyzer

Untuk melihat sumber daya yang ditentukan oleh IAM Access Analyzer, lihat Jenis sumber daya yang ditentukan oleh IAM Access Analyzer di Referensi Otorisasi Layanan.

Izin layanan IAM Access Analyzer yang diperlukan

IAMAccess Analyzer menggunakan peran terkait layanan () SLR bernama. `AWSServiceRoleForAccessAnalyzer` Ini SLR memberikan layanan akses hanya-baca untuk menganalisis AWS sumber daya dengan kebijakan berbasis sumber daya dan menganalisis akses yang tidak digunakan atas nama Anda. Layanan membuat peran di akun Anda dalam skenario berikut:

- Anda membuat analisa akses eksternal dengan akun Anda sebagai zona kepercayaan.
- Anda membuat penganalisis akses yang tidak terpakai dengan akun Anda sebagai akun yang dipilih.

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer.

> **Note**
>
> IAMAccess Analyzer adalah Regional. Untuk akses eksternal, Anda harus mengaktifkan IAM Access Analyzer di setiap Wilayah secara independen.

Untuk akses yang tidak digunakan, temuan untuk penganalisis tidak berubah berdasarkan Wilayah. Membuat analyzer di setiap Wilayah di mana Anda memiliki sumber daya tidak diperlukan.

Dalam beberapa kasus, setelah Anda membuat akses eksternal atau penganalisis akses yang tidak digunakan di IAM Access Analyzer, halaman atau dasbor temuan dimuat tanpa temuan atau ringkasan. Hal ini mungkin disebabkan oleh penundaan pada konsol untuk mengisi temuan Anda. Anda mungkin perlu menyegarkan browser secara manual atau memeriksa kembali nanti untuk melihat temuan atau ringkasan Anda. Jika Anda masih tidak melihat temuan untuk penganalisis akses eksternal, itu karena Anda tidak memiliki sumber daya yang didukung di akun Anda yang dapat diakses oleh entitas eksternal. Jika kebijakan yang memberikan akses ke entitas eksternal diterapkan ke sumber daya, IAM Access Analyzer akan menghasilkan temuan.

Note

Untuk penganalisis akses eksternal, diperlukan waktu hingga 30 menit setelah kebijakan diubah untuk IAM Access Analyzer untuk menganalisis sumber daya dan kemudian menghasilkan temuan akses eksternal baru atau memperbarui temuan yang ada untuk akses ke sumber daya. Untuk penganalisis akses eksternal dan tidak terpakai, pembaruan untuk temuan mungkin tidak segera tercermin di dasbor.

Izin IAM Access Analyzer yang diperlukan untuk melihat dasbor temuan

Untuk melihat dasbor temuan IAM Access Analyzer, akun yang Anda gunakan harus diberikan akses untuk melakukan tindakan yang diperlukan berikut:

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

Untuk melihat semua tindakan yang ditentukan oleh IAM Access Analyzer, lihat Tindakan yang ditentukan oleh IAM Access Analyzer di Referensi Otorisasi Layanan.

Mengaktifkan IAM Access Analyzer

Untuk membuat analisa akses eksternal dengan Akun AWS sebagai zona kepercayaan

Untuk mengaktifkan penganalisis akses eksternal di Wilayah, Anda harus membuat penganalisis di Wilayah tersebut. Anda harus membuat penganalisis akses eksternal di setiap Wilayah tempat Anda ingin memantau akses ke sumber daya Anda.

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses.
3. Pilih Pengaturan Analyzer.
4. Pilih Membuat penganalisis.
5. Di bagian Analisis, pilih Analisis akses eksternal.
6. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
7. Masukkan nama penganalisis.
8. Pilih Current Akun AWS sebagai zona kepercayaan untuk analyzer.

> **Note**
>
> Jika akun Anda bukan akun AWS Organizations manajemen atau akun administrator yang didelegasikan, Anda hanya dapat membuat satu penganalisis dengan akun Anda sebagai zona kepercayaan.

9. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Kirim.

Saat Anda membuat penganalisis akses eksternal untuk mengaktifkan IAM Access Analyzer, peran terkait layanan bernama akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Untuk membuat analisa akses eksternal dengan organisasi sebagai zona kepercayaan

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses.
3. Pilih Pengaturan Analyzer.
4. Pilih Membuat penganalisis.

5. Di bagian Analisis, pilih Analisis akses eksternal.
6. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
7. Masukkan nama penganalisis.
8. Pilih Organisasi saat ini sebagai zona kepercayaan untuk penganalisis.
9. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Kirim.

Saat Anda membuat penganalisis akses eksternal dengan organisasi sebagai zona kepercayaan, nama peran terkait layanan akan `AWSServiceRoleForAccessAnalyzer` dibuat di setiap akun organisasi Anda.

Untuk membuat penganalisis akses yang tidak digunakan untuk akun saat ini

Gunakan prosedur berikut untuk membuat penganalisis akses yang tidak terpakai untuk satu. Akun AWS Untuk akses yang tidak digunakan, temuan untuk penganalisis tidak berubah berdasarkan Wilayah. Membuat analyzer di setiap Wilayah di mana Anda memiliki sumber daya tidak diperlukan.

IAMAccess Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per bulan per penganalisis. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses.
3. Pilih Pengaturan Analyzer.
4. Pilih Membuat penganalisis.
5. Di bagian Analisis, pilih Analisis akses yang tidak digunakan.
6. Masukkan nama penganalisis.
7. Untuk periode Tracking, masukkan jumlah hari untuk menghasilkan temuan untuk izin yang tidak digunakan. Misalnya, jika Anda memasukkan 90 hari, penganalisis akan menghasilkan temuan untuk IAM entitas dalam akun yang dipilih untuk setiap izin yang belum digunakan dalam 90 hari atau lebih sejak pemindaian terakhir penganalisis. Anda dapat memilih nilai antara 1 dan 180 hari.
8. Untuk Akun yang Dipilih, pilih Saat Ini Akun AWS.

> **Note**
>
> Jika akun Anda bukan akun AWS Organizations manajemen atau akun administrator yang didelegasikan, Anda hanya dapat membuat satu penganalisis dengan akun Anda sebagai akun yang dipilih.

9. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Kirim.

Saat Anda membuat penganalisis akses yang tidak digunakan untuk mengaktifkan IAM Access Analyzer, nama peran terkait layanan akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Untuk membuat penganalisis akses yang tidak terpakai dengan organisasi saat ini

Gunakan prosedur berikut untuk membuat penganalisis akses yang tidak terpakai bagi organisasi untuk meninjau secara terpusat semua Akun AWS dalam organisasi. Untuk analisis akses yang tidak digunakan, temuan untuk penganalisis tidak berubah berdasarkan Wilayah. Membuat analyzer di setiap Wilayah di mana Anda memiliki sumber daya tidak diperlukan.

IAMAccess Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per bulan per penganalisis. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

> **Note**
>
> Jika akun anggota dihapus dari organisasi, penganalisis akses yang tidak digunakan akan berhenti menghasilkan temuan baru dan memperbarui temuan yang ada untuk akun tersebut setelah 24 jam. Temuan yang terkait dengan akun anggota yang dihapus dari organisasi akan dihapus secara permanen setelah 90 hari.

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses.
3. Pilih Pengaturan Analyzer.
4. Pilih Membuat penganalisis.

5. Di bagian Analisis, pilih Analisis akses yang tidak digunakan.
6. Masukkan nama penganalisis.
7. Untuk periode Tracking, masukkan jumlah hari untuk menghasilkan temuan untuk izin yang tidak digunakan. Misalnya, jika Anda memasukkan 90 hari, penganalisis akan menghasilkan temuan untuk IAM entitas dalam akun organisasi yang dipilih untuk izin apa pun yang belum digunakan dalam 90 hari atau lebih sejak pemindaian terakhir penganalisis. Anda dapat memilih nilai antara 1 dan 180 hari.
8. Untuk Akun yang dipilih, pilih Organisasi saat ini sebagai akun yang dipilih untuk penganalisis.
9. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Kirim.

Saat Anda membuat penganalisis akses yang tidak digunakan untuk mengaktifkan IAM Access Analyzer, nama peran terkait layanan akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

IAMAkses status Analyzer

Untuk melihat status analyzer Anda, pilih Penganalisis. Penganalisis yang dibuat untuk organisasi atau akun dapat memiliki status sebagai berikut:

Status	Deskripsi
Aktif	Untuk penganalisis akses eksternal, penganalisis secara aktif memantau sumber daya dalam zona kepercayaannya. Analyzer secara aktif menghasilkan temuan baru dan memperbarui temuan yang ada. Untuk penganalisis akses yang tidak digunakan, penganalisis secara aktif memantau akses yang tidak digunakan dalam organisasi yang dipilih atau Akun AWS dalam periode pelacakan yang ditentukan. Analyzer secara aktif menghasilkan temuan baru dan memperbarui temuan yang ada.

Status	Deskripsi
Membuat	Pembuatan penganalisis masih dalam proses. Penganalisis menjadi aktif setelah pembuatan selesai.
Nonaktif	Penganalisis dinonaktifkan karena tindakan yang diambil oleh AWS Organizations administrator. Misalnya, menghapus akun analyzer sebagai administrator yang didelegasikan untuk IAM Access Analyzer. Ketika penganalisis dalam keadaan dinonaktifkan, itu tidak menghasilkan temuan baru atau memperbarui temuan yang ada.
Failed	Pembuatan penganalisis gagal karena masalah konfigurasi. Penganalisis tidak akan menghasilkan temuan apa pun. Hapus penganalisis dan buat penganalisis baru.

Lihat dasbor temuan IAM Access Analyzer

AWS Identity and Access Management Access Analyzer mengatur akses eksternal dan temuan akses yang tidak digunakan ke dalam dasbor ringkasan visual. Dasbor membantu Anda mendapatkan visibilitas ke dalam penggunaan izin yang efektif dalam skala besar dan mengidentifikasi akun yang perlu diperhatikan. Anda dapat menggunakan dasbor untuk meninjau temuan berdasarkan AWS organisasi, akun, dan jenis temuan.

Untuk temuan akses eksternal:

- Dasbor menyoroti pemisahan antara akses publik dan temuan akses lintas akun.
- Dasbor menyediakan rincian temuan berdasarkan jenis sumber daya.

Untuk temuan akses yang tidak digunakan:

- Dasbor menyoroti Akun AWS dengan temuan akses yang paling tidak digunakan.
- Dasbor memberikan rincian temuan berdasarkan jenis.

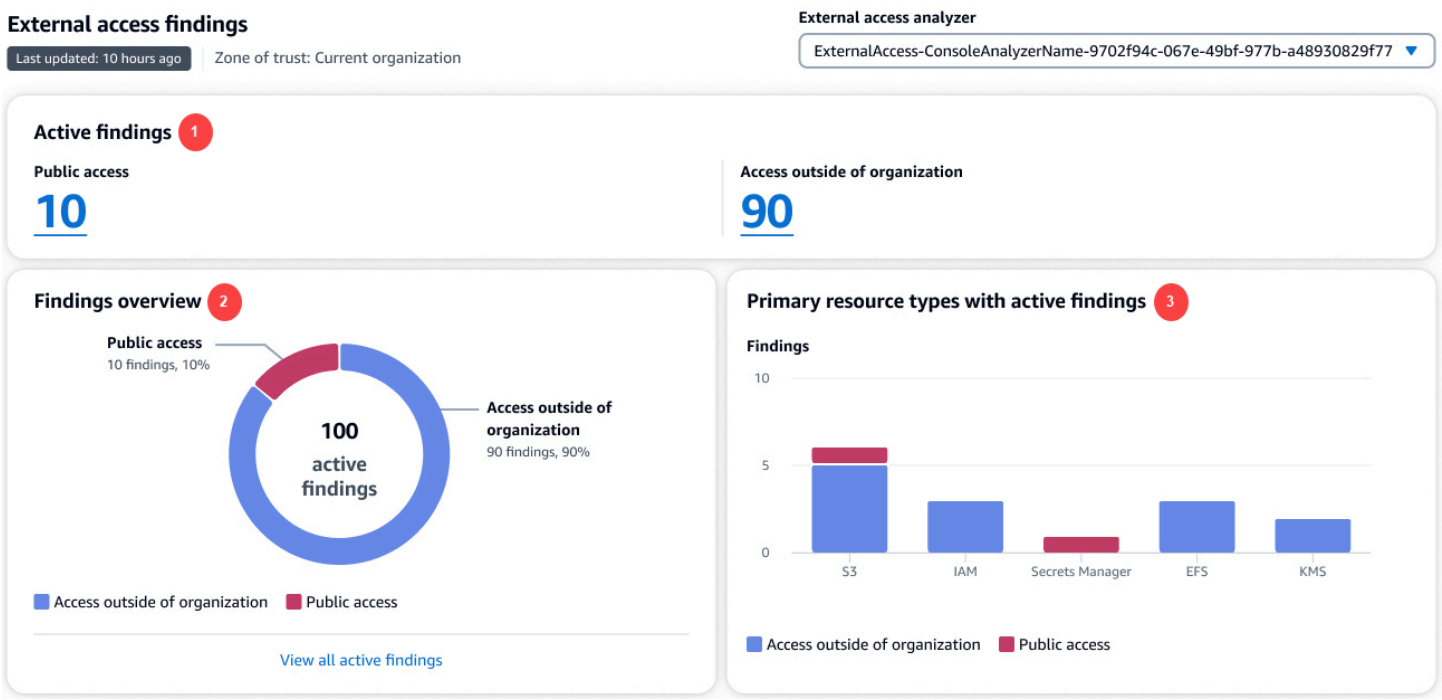
Setelah Anda membuat penganalisis untuk akses eksternal atau tidak terpakai, IAM Access Analyzer secara otomatis menambahkan temuan baru ke dasbor yang relevan. Ini memungkinkan Anda untuk mengidentifikasi dan memprioritaskan area dengan masalah keamanan paling banyak.

Dasbor ringkasan memberi Anda tampilan tingkat tinggi tentang masalah akses yang terdeteksi oleh IAM Access Analyzer di seluruh lingkungan Anda. AWS Anda kemudian dapat menelusuri temuan individu untuk menyelidiki lebih lanjut dan mengambil tindakan yang tepat untuk menyelesaikannya.

Untuk melihat dasbor ringkasan untuk penganalisis akses eksternal

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, dasbor ringkasan dapat memakan waktu untuk mencerminkan pembaruan temuan.

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses. Jendela Ringkasan ditampilkan.
3. Pilih analyzer dari dropdown External access analyzer. Ringkasan temuan untuk penganalisis ditampilkan di bagian Temuan akses eksternal.



Pada gambar sebelumnya, dasbor temuan akses eksternal terlihat dari dalam halaman Ringkasan:

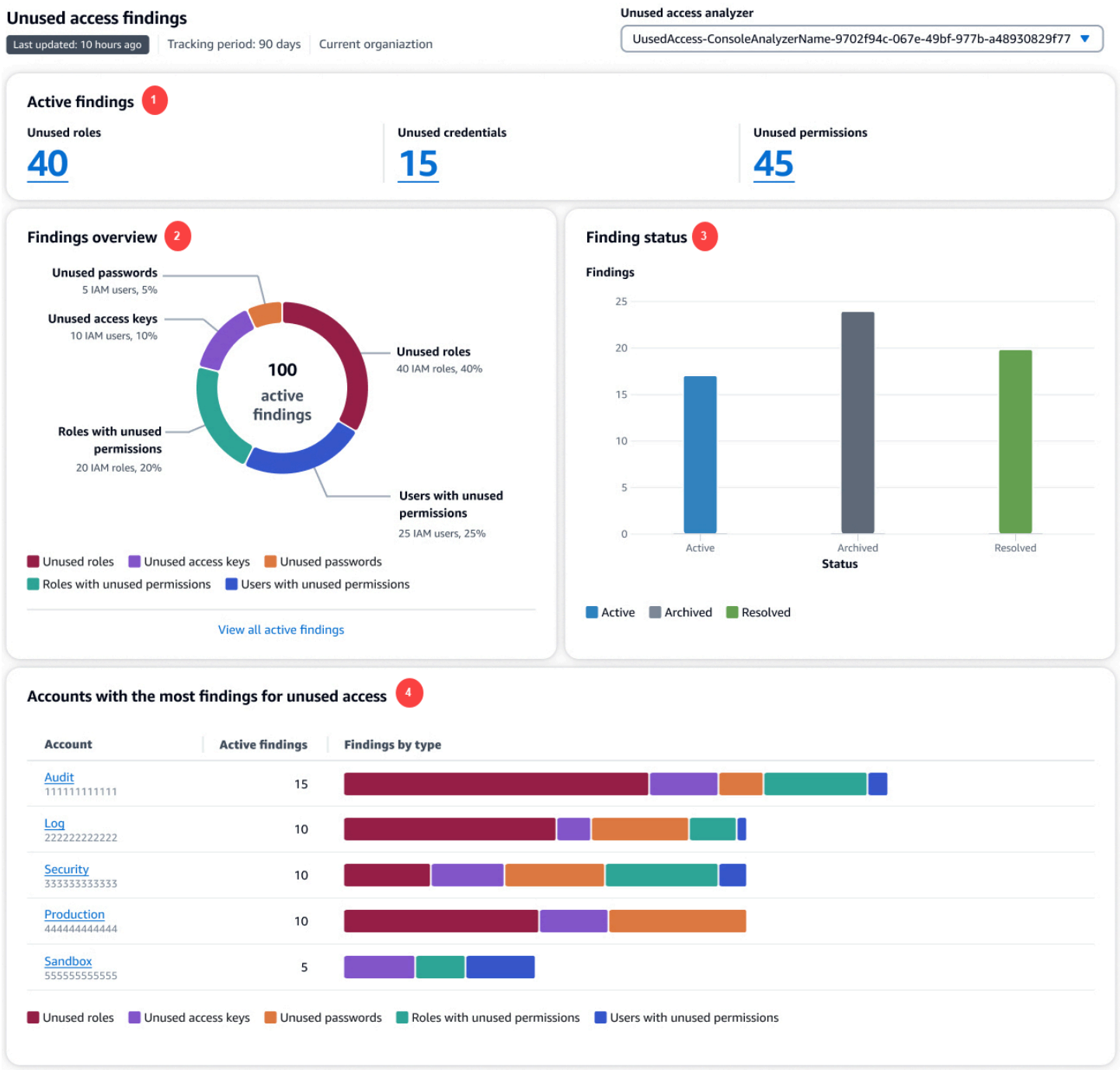
1. Bagian temuan aktif mencakup jumlah temuan aktif untuk akses publik dan jumlah temuan aktif yang menyediakan akses di luar akun atau organisasi. Pilih nomor untuk mencantumkan semua temuan aktif dari setiap jenis.
2. Bagian ikhtisar Temuan mencakup rincian jenis temuan aktif. Pilih Lihat semua temuan aktif untuk daftar lengkap temuan aktif untuk akun atau organisasi penganalisis.
3. Jenis sumber daya primer dengan bagian temuan aktif mencakup rincian jenis sumber daya primer dengan temuan aktif. Informasi ini membantu Anda memprioritaskan temuan untuk sumber daya utama terlebih dahulu. Misalnya, Amazon S3, DynamoDB, dan. AWS KMS Ini bukan daftar lengkap dari setiap jenis sumber daya. Penganalisis Anda mungkin memiliki temuan aktif untuk jenis sumber daya yang tidak tercantum di bagian ini.

Untuk melihat dasbor ringkasan untuk penganalisis akses yang tidak digunakan

IAMAccess Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, berdasarkan jumlah pengguna dan peran, dasbor ringkasan dapat memakan waktu untuk mencerminkan pembaruan temuan.

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses. Jendela Ringkasan ditampilkan.
3. Pilih penganalisis dari dropdown penganalisis akses yang tidak digunakan. Ringkasan temuan untuk penganalisis ditampilkan di bagian Temuan akses yang tidak digunakan.



Pada gambar sebelumnya, dasbor temuan akses eksternal terlihat dari dalam halaman Ringkasan:

1. Bagian temuan aktif mencakup jumlah temuan aktif untuk peran yang tidak digunakan, kredensi yang tidak digunakan, dan izin yang tidak digunakan di akun atau organisasi Anda. Kredensi yang tidak digunakan mencakup kunci akses yang tidak digunakan dan temuan kata sandi yang tidak digunakan. Izin yang tidak digunakan mencakup pengguna dan peran dengan izin yang tidak digunakan. Pilih nomor untuk mencantumkan semua temuan aktif dari setiap jenis.

2. Bagian ikhtisar Temuan mencakup rincian jenis temuan aktif. Pilih Lihat semua temuan aktif untuk daftar lengkap temuan aktif untuk akun atau organisasi penganalisis.
3. Bagian Status pencarian mencakup rincian status temuan (Aktif, Diarsipkan, dan Terselesaikan) untuk akun atau organisasi Anda.
4. Bagian Akun dengan temuan terbanyak untuk akses yang tidak digunakan hanya ditampilkan jika akun yang dipilih dari penganalisis akses Anda yang tidak digunakan berada di tingkat organisasi. Ini mencakup rincian akun di organisasi Anda dengan temuan paling aktif. Ini bukan daftar lengkap dari setiap akun di organisasi Anda. Analyzer Anda mungkin memiliki temuan aktif untuk akun lain yang tidak tercantum di bagian ini.

Tinjau temuan IAM Access Analyzer

Setelah Anda mengaktifkan IAM Access Analyzer, langkah selanjutnya adalah meninjau temuan apa pun untuk menentukan apakah akses yang diidentifikasi dalam temuan itu disengaja atau tidak disengaja. Anda juga dapat meninjau temuan untuk menentukan temuan serupa untuk akses yang dimaksudkan, dan kemudian membuat aturan arsip untuk mengarsipkan temuan tersebut secara otomatis. Anda juga dapat meninjau temuan yang diarsipkan dan diselesaikan.

Anda harus meninjau semua temuan di akun Anda untuk menentukan apakah akses eksternal atau tidak terpakai diharapkan dan disetujui. Jika akses eksternal atau tidak terpakai yang diidentifikasi dalam temuan diharapkan, Anda dapat mengarsipkan temuan tersebut. Saat Anda mengarsipkan temuan, statusnya diubah menjadi Diarsipkan, dan temuan tersebut dihapus dari daftar temuan aktif. Temuan tidak dihapus. Anda dapat melihat temuan yang diarsipkan kapan saja. Pelajari semua temuan di akun Anda hingga Anda tidak memiliki temuan aktif. Setelah Anda mencapai nol temuan, Anda tahu bahwa setiap temuan Aktif baru yang dihasilkan berasal dari perubahan baru-baru ini di lingkungan Anda.

Untuk meninjau temuan

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses.
3. Dasbor temuan ditampilkan. Pilih temuan aktif untuk penganalisis akses eksternal atau tidak terpakai Anda.

   Untuk informasi lebih lanjut tentang melihat dasbor temuan, lihatLihat dasbor temuan IAM Access Analyzer.

> **Note**
>
> Temuan ditampilkan hanya jika Anda memiliki izin untuk melihat temuan untuk penganalisis.

Semua temuan ditampilkan untuk penganalisis. Untuk melihat temuan lain yang dihasilkan oleh penganalisis, pilih jenis temuan yang sesuai dari tarik-turun Status:

- Pilih Aktif untuk melihat semua temuan aktif yang dihasilkan oleh penganalisis.
- Pilih Diarsipkan untuk melihat hanya temuan yang dihasilkan oleh penganalisis yang telah diarsipkan. Untuk mempelajari informasi lebih lanjut, lihat Arsipkan IAM temuan Access Analyzer.
- Pilih Diselesaikan untuk melihat hanya temuan yang dihasilkan oleh penganalisis yang telah diselesaikan. Saat Anda memulihkan masalah yang menghasilkan temuan, status temuan diubah menjadi Terselesaikan.

> **Important**
>
> Temuan yang diselesaikan dihapus 90 hari setelah pembaruan terakhir temuan. Temuan aktif dan diarsipkan tidak akan dihapus kecuali jika Anda menghapus penganalisis yang menghasilkannya.

- Pilih Semua untuk melihat semua temuan dengan semua status yang dihasilkan oleh penganalisis.

Temuan akses eksternal

Pilih Akses eksternal dan kemudian pilih penganalisis akses eksternal dari menu tarik-turun View analyzer. Halaman Temuan untuk penganalisis akses eksternal menampilkan detail berikut tentang sumber daya bersama dan pernyataan kebijakan yang menghasilkan temuan:

Menemukan ID

ID unik yang ditetapkan untuk temuan. Pilih ID temuan untuk menampilkan detail tambahan tentang sumber daya dan pernyataan kebijakan yang menghasilkan temuan tersebut.

Sumber Daya

Jenis dan nama parsial dari sumber daya yang memiliki kebijakan yang diterapkan yang memberikan akses ke entitas eksternal yang tidak berada di zona kepercayaan Anda.

Akun pemilik sumber daya

Kolom ini ditampilkan hanya jika Anda menggunakan sebuah organisasi sebagai zona kepercayaan. Akun dalam organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.

Prinsipal eksternal

Penanggung jawab, tidak di dalam zona kepercayaan Anda yang diberikan akses oleh kebijakan yang dianalisis. Nilai yang valid meliputi:

- Akun AWS— Semua prinsipal dalam daftar Akun AWS dengan izin dari administrator akun itu dapat mengakses sumber daya.
- Prinsipal apa pun — Semua kepala sekolah Akun AWS yang memenuhi persyaratan yang termasuk dalam kolom Ketentuan memiliki izin untuk mengakses sumber daya. Misalnya, jika a VPC terdaftar, itu berarti bahwa setiap prinsipal di akun apa pun yang memiliki izin untuk mengakses yang terdaftar VPC dapat mengakses sumber daya.
- Pengguna kanonik — Semua kepala sekolah Akun AWS dengan ID pengguna kanonik yang terdaftar memiliki izin untuk mengakses sumber daya.
- IAMperan — IAM Peran yang terdaftar memiliki izin untuk mengakses sumber daya.
- IAMpengguna — Pengguna yang terdaftar IAM memiliki izin untuk mengakses sumber daya.

Kondisi

Ketentuan dari pernyataan kebijakan yang memberikan akses. Misalnya, jika bidang Kondisi menyertakan Sumber VPC, itu berarti sumber daya dibagikan dengan prinsipal yang memiliki akses ke VPC daftar. Ketentuan dapat bersifat global atau khusus layanan. Kunci ketentuan global memiliki `aws:` awalan.

Dibagikan melalui

Bidang Dibagikan melalui menunjukkan bagaimana akses yang menghasilkan temuan diberikan. Nilai yang valid meliputi:

- Kebijakan Bucket — Kebijakan bucket yang dilampirkan pada bucket Amazon S3.
- Daftar kontrol akses — Daftar kontrol akses (ACL) yang dilampirkan ke bucket Amazon S3.
- Titik akses — Titik akses atau titik akses multi-wilayah yang terkait dengan bucket Amazon S3. Titik akses ditampilkan dalam rincian Temuan. ARN

Tingkat akses

Tingkat akses yang diberikan kepada entitas eksternal oleh tindakan dalam kebijakan berbasis sumber daya. Lihat detail temuan untuk informasi lebih lanjut. Nilai tingkat akses mencakup hal berikut:

- Daftar – Izin untuk mencantumkan sumber daya di dalam layanan untuk menentukan apakah ada objek. Tindakan dengan tingkat akses ini dapat mencantumkan objek tetapi tidak dapat melihat isi sumber daya.
- Baca – Izin untuk membaca, namun tidak mengedit konten dan atribut sumber daya dalam layanan.
- Tulis – Izin untuk membuat, menghapus, atau memodifikasi sumber daya dalam layanan.
- Izin – Izin untuk memberikan atau mengubah izin sumber daya dalam layanan.
- Penandaan – Izin untuk melakukan tindakan yang hanya mengubah status tanda sumber daya.

Diperbarui

Stempel waktu untuk pembaruan terbaru ke status temuan, atau waktu dan tanggal temuan dihasilkan jika tidak ada pembaruan yang dilakukan.

> **Note**
>
> Diperlukan waktu hingga 30 menit setelah kebijakan diubah agar IAM Access Analyzer kembali menganalisis sumber daya dan kemudian memperbarui temuannya.

Status

Status temuan, salah satu Aktif, Diarsipkan, atau Diselesaikan.

Temuan akses yang tidak digunakan

IAMAccess Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Pilih Akses yang tidak digunakan dan kemudian pilih penganalisis akses yang tidak terpakai dari menu tarik-turun View analyzer. Halaman Temuan untuk penganalisis akses yang tidak digunakan menampilkan detail berikut tentang IAM entitas yang menghasilkan temuan:

Menemukan ID

ID unik yang ditetapkan untuk temuan. Pilih ID temuan untuk menampilkan detail tambahan tentang IAM entitas yang menghasilkan temuan.

Menemukan jenis

Jenis pencarian akses yang tidak digunakan: Kunci akses yang tidak digunakan, Kata sandi yang tidak digunakan, Izin yang tidak digunakan, atau Peran yang tidak digunakan.

IAMentitas

IAMEntitas dilaporkan dalam temuan. Ini bisa menjadi IAM pengguna atau peran.

Akun AWS ID

Kolom ini hanya ditampilkan jika Anda mengatur penganalisis untuk semua yang Akun AWS ada di organisasi. Akun AWS Dalam organisasi yang memiliki IAM entitas dilaporkan dalam temuan.

Terakhir diperbarui

Terakhir kali IAM entitas melaporkan dalam temuan diperbarui, atau ketika entitas dibuat jika tidak ada pembaruan yang dilakukan.

Status

Status temuan (Aktif, Diarsipkan, atau Diselesaikan).

Temuan Filter IAM Access Analyzer

Pemfilteran default untuk halaman temuan adalah untuk menampilkan semua temuan. Untuk melihat temuan aktif, pilih status Aktif dari tarik-turun Status. Untuk melihat temuan yang diarsipkan, pilih Status yang diarsipkan dari menu tarik-turun Status. Saat pertama kali mulai menggunakan IAM Access Analyzer, tidak ada temuan yang diarsipkan.

Gunakan filter untuk menampilkan hanya temuan yang memenuhi kriteria properti yang ditentukan. Untuk membuat filter, pilih properti yang akan difilter, lalu pilih apakah properti sama atau berisi nilai, lalu masukkan atau pilih nilai properti untuk difilter. Misalnya, untuk membuat filter yang hanya menampilkan temuan tertentu Akun AWS, pilih AWS Akun untuk properti, lalu pilih AWS Akun =, lalu masukkan nomor akun Akun AWS yang ingin Anda lihat temuannya.

Untuk daftar kunci filter yang dapat Anda gunakan untuk membuat atau memperbarui aturan arsip, lihat Kunci filter IAM Access Analyzer.

Memfilter temuan akses eksternal

Untuk menyaring temuan akses eksternal

1. Pilih Akses eksternal dan kemudian pilih analyzer di menu tarik-turun View analyzer.
2. Pilih kotak pencarian untuk menampilkan daftar properti yang tersedia.
3. Pilih properti yang akan digunakan untuk memfilter temuan yang ditampilkan.
4. Pilih nilai yang sesuai dengan properti. Hanya temuan dengan nilai tersebut dalam temuan yang ditampilkan.

   Misalnya, pilih Resource sebagai properti, lalu pilih Resource:, lalu ketik sebagian atau seluruh nama bucket, lalu tekan Enter. Hanya temuan untuk bucket yang cocok dengan kriteria filter yang ditampilkan. Untuk membuat filter yang hanya menampilkan temuan untuk sumber daya yang memungkinkan akses publik, Anda dapat memilih properti Akses publik, lalu pilih Akses publik =, lalu pilih Akses publik = true.

Anda dapat menambahkan properti tambahan untuk memfilter temuan yang ditampilkan lebih lanjut. Saat Anda menambahkan properti tambahan, hanya temuan yang cocok dengan semua ketentuan dalam filter yang ditampilkan. Menetapkan filter untuk menampilkan temuan yang cocok dengan satu properti ATAU properti lain tidak didukung. Pilih Hapus filter untuk menghapus filter apa pun yang telah Anda tetapkan dan menampilkan semua temuan dengan status yang ditentukan untuk penganalisis Anda.

Beberapa bidang hanya ditampilkan saat Anda melihat temuan untuk penganalisis dengan organisasi sebagai zona kepercayaannya.

Properti berikut tersedia untuk menentukan filter:

- Akses publik – Untuk memfilter berdasarkan temuan untuk sumber daya yang memungkinkan akses publik, filter berdasarkan Akses publik, lalu pilih Akses publik: benar.
- Sumber Daya – Untuk memfilter berdasarkan sumber daya, ketik semua atau sebagian nama sumber daya.
- Jenis Sumber Daya – Untuk memfilter berdasarkan jenis sumber daya, pilih jenis dari daftar yang ditampilkan.
- Akun Pemilik Sumber Daya — Gunakan properti ini untuk memfilter berdasarkan akun di organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.

- AWS Akun — Gunakan properti ini untuk memfilter dengan akses Akun AWS yang diberikan di bagian Utama dari pernyataan kebijakan. Untuk memfilter berdasarkan Akun AWS, ketik semua atau sebagian dari Akun AWS ID 12 digit, atau semua atau sebagian ARN dari akun lengkap AWS pengguna eksternal atau peran yang memiliki akses ke sumber daya di akun saat ini.
- Pengguna Canonical — Untuk memfilter menurut pengguna kanonik, ketikkan ID pengguna kanonik seperti yang ditentukan untuk bucket Amazon S3. Untuk mempelajari selengkapnya, lihat Pengidentifikasi Akun AWS.
- Pengguna Federasi — Untuk memfilter berdasarkan pengguna federasi, ketik semua atau sebagian dari identitas ARN federasi. Untuk mempelajari, lihat Penyedia dan Gabungan Identitas.
- Menemukan ID — Untuk memfilter dengan menemukan ID, ketik semua atau sebagian dari ID pencarian.
- Kesalahan — Untuk memfilter berdasarkan jenis kesalahan, pilih Akses Ditolak atau Kesalahan Internal.
- Principal ARN — Gunakan properti ini untuk memfilter prinsipal (IAMpengguna, peran, atau grup) yang digunakan dalam aws: PrincipalArn condition key. ARN Untuk memfilter menurut PrincipalARN, ketik semua atau sebagian IAM pengguna, peran, atau grup dari eksternal yang Akun AWS dilaporkan dalam temuan. ARN
- Principal OrgID — Untuk memfilter berdasarkan Principal OrGid, ketik semua atau sebagian ID organisasi yang terkait dengan prinsipal eksternal yang termasuk dalam AWS organisasi yang ditentukan sebagai kondisi dalam temuan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Principal OrgPaths — Untuk memfilter menurut Principal OrgPaths, ketikkan semua atau sebagian ID untuk AWS organisasi atau unit organisasi (OU) yang memungkinkan akses ke semua kepala sekolah eksternal yang merupakan anggota akun dari organisasi tertentu atau OU sebagai syarat dalam kebijakan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Akun Sumber — Untuk memfilter di Akun Sumber, ketik semua atau sebagian Akun AWS ID yang terkait dengan sumber daya, seperti yang digunakan dalam beberapa izin lintas layanan. AWS Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Sumber ARN — Untuk memfilter berdasarkan SumberARN, ketik semua atau sebagian yang ARN ditentukan sebagai kondisi dalam temuan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- IP Sumber – Untuk memfilter berdasarkan IP Sumber, ketikkan semua atau sebagian alamat IP yang memungkinkan akses entitas eksternal ke sumber daya dalam akun saat menggunakan alamat IP yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.

- Sumber VPC - Untuk memfilter berdasarkan SumberVPC, ketik semua atau sebagian VPC ID yang memungkinkan entitas eksternal mengakses sumber daya di akun saat menggunakan yang ditentukanVPC. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Sumber OrgID — Untuk memfilter berdasarkan Sumber OrgID, ketik semua atau sebagian ID organisasi yang terkait dengan sumber daya, seperti yang digunakan dalam beberapa izin lintas layanan di. AWS Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Sumber OrgPaths — Untuk memfilter berdasarkan Sumber OrgPaths, ketik semua atau sebagian unit organisasi (OU) yang terkait dengan sumber daya, seperti yang digunakan dalam beberapa izin lintas layanan di AWS. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- User ID — Untuk memfilter berdasarkan User ID, ketik semua atau sebagian ID pengguna IAM pengguna dari eksternal Akun AWS yang diizinkan mengakses sumber daya di akun saat ini. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- KMSID Kunci — Untuk memfilter berdasarkan ID KMS kunci, ketik semua atau sebagian ID kunci untuk KMS kunci yang ditentukan sebagai kondisi untuk akses objek Amazon AWS KMS S3 yang dienkripsi di akun Anda saat ini.
- Google Audience — Untuk memfilter menurut Google Audience, ketik semua atau sebagian ID aplikasi Google yang ditentukan sebagai syarat untuk akses IAM peran di akun Anda saat ini. Untuk mempelajari lebih lanjut, lihat IAMdan AWS STS kondisikan kunci konteks.
- Pemirsa Cognito — Untuk memfilter menurut audiens Amazon Cognito, ketik semua atau sebagian ID kumpulan identitas Amazon Cognito yang ditentukan sebagai syarat IAM untuk akses peran di akun Anda saat ini. Untuk mempelajari lebih lanjut, lihat IAMdan AWS STS kondisikan kunci konteks.
- Akun Penelepon — Akun AWS ID akun yang memiliki atau berisi entitas pemanggil, seperti IAM peran, pengguna, atau pengguna root akun. Ini digunakan oleh panggilan layanan AWS KMS. Untuk memfilter berdasarkan akun penelepon, ketik semua atau sebagian Akun AWS ID.
- ID Aplikasi Facebook — Untuk memfilter berdasarkan ID Aplikasi Facebook, ketik semua atau sebagian ID aplikasi Facebook (atau ID situs) yang ditentukan sebagai syarat untuk mengizinkan Login dengan federasi Facebook akses ke IAM peran di akun Anda saat ini. Untuk mempelajari lebih lanjut, lihat bagian id dalam IAMdan AWS STS kondisi kunci konteks.
- ID Aplikasi Amazon — Untuk memfilter berdasarkan ID Aplikasi Amazon, ketik semua atau sebagian ID aplikasi Amazon (atau ID situs) yang ditentukan sebagai syarat untuk mengizinkan Login with Amazon federasi akses ke IAM peran di akun Anda saat ini. Untuk mempelajari lebih lanjut, lihat bagian id dalam IAMdan AWS STS kondisi kunci konteks.

- Token Sumber Peristiwa Lambda – Untuk memfilter Token Sumber Peristiwa Lambda yang diteruskan dengan integrasi Alexa, ketikkan semua atau sebagian string token.

Memfilter temuan akses yang tidak digunakan

Untuk menyaring temuan akses yang tidak digunakan

1. Pilih Akses yang tidak digunakan dan kemudian pilih penganalisis di menu tarik-turun View analyzer.
2. Pilih kotak pencarian untuk menampilkan daftar properti yang tersedia.
3. Pilih properti yang akan digunakan untuk memfilter temuan yang ditampilkan.
4. Pilih nilai yang sesuai dengan properti. Hanya temuan dengan nilai tersebut dalam temuan yang ditampilkan.

   Misalnya, pilih Jenis temuan sebagai properti, lalu pilih Jenis temuan =, lalu pilih Jenis temuan = U nusedIAMRole, Hanya temuan dengan tipe U nusedIAMRole yang ditampilkan.

Anda dapat menambahkan properti tambahan untuk memfilter temuan yang ditampilkan lebih lanjut. Saat Anda menambahkan properti tambahan, hanya temuan yang cocok dengan semua ketentuan dalam filter yang ditampilkan. Menetapkan filter untuk menampilkan temuan yang cocok dengan satu properti ATAU properti lain tidak didukung. Pilih Hapus filter untuk menghapus filter apa pun yang telah Anda tetapkan dan menampilkan semua temuan dengan status yang ditentukan untuk penganalisis Anda.

Bidang berikut hanya ditampilkan saat Anda melihat temuan untuk penganalisis yang memantau akses yang tidak digunakan:

- Jenis temuan — Untuk memfilter berdasarkan jenis pencarian, filter berdasarkan jenis Temuan dan kemudian pilih jenis temuan.
- Sumber Daya – Untuk memfilter berdasarkan sumber daya, ketik semua atau sebagian nama sumber daya.
- Jenis Sumber Daya – Untuk memfilter berdasarkan jenis sumber daya, pilih jenis dari daftar yang ditampilkan.
- Akun Pemilik Sumber Daya — Gunakan properti ini untuk memfilter berdasarkan akun di organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.
- Mencari id — Untuk memfilter dengan mencari ID, ketik semua atau sebagian dari ID pencarian.

Arsipkan IAM temuan Access Analyzer

Ketika Anda mendapatkan temuan untuk akses ke sumber daya yang disengaja, Anda dapat mengarsipkan temuan. Misalnya, pencarian akses eksternal untuk IAM peran yang digunakan oleh beberapa pengguna untuk alur kerja yang disetujui atau pencarian akses yang tidak digunakan untuk kunci akses yang mungkin masih diperlukan. Ketika Anda mengarsipkan temuan, itu dihapus dari daftar temuan aktif. Temuan yang diarsipkan tidak dihapus. Anda dapat memfilter halaman Temuan untuk menampilkan temuan yang diarsipkan, dan membatalkan arsipkannya kapan saja.

Untuk mengarsipkan temuan dari laman Temuan

1. Pilih kotak centang di samping satu atau beberapa temuan untuk diarsipkan.
2. Pilih Tindakan dan kemudian pilih Arsip.

   Konfirmasi ditampilkan di bagian atas layar.

Untuk mengarsipkan temuan dari halaman Detail Temuan

1. Pilih ID Temuan untuk mengarsip temuan.

2. Pilih Arsipkan.

   Konfirmasi ditampilkan di bagian atas layar.

Untuk menghapus arsip temuan, ulangi langkah sebelumnya, tetapi pilih Batalkan Arsip alih-alih Arsipkan. Saat Anda menghapus arsip temuan, status diatur ke Aktif.

Selesaikan IAM temuan Access Analyzer

Menyelesaikan temuan akses eksternal

Untuk menyelesaikan temuan akses eksternal yang dihasilkan dari akses yang tidak diinginkan, Anda harus mengubah pernyataan kebijakan untuk menghapus izin yang memungkinkan akses ke sumber daya yang diidentifikasi.

Untuk temuan yang terkait dengan bucket Amazon S3, gunakan konsol Amazon S3 untuk mengonfigurasi izin di bucket.

Untuk IAM peran, gunakan IAM konsol untuk mengubah kebijakan kepercayaan untuk IAM peran yang terdaftar.

Untuk sumber daya lain yang didukung, gunakan konsol untuk mengubah pernyataan kebijakan yang menghasilkan temuan yang dihasilkan.

Setelah membuat perubahan untuk menyelesaikan pencarian akses eksternal, seperti memodifikasi kebijakan yang diterapkan ke IAM peran, IAM Access Analyzer akan memindai sumber daya lagi. Jika sumber daya tidak lagi dibagikan di luar zona kepercayaan Anda, status temuan diubah menjadi Terselesaikan. Temuan ini kemudian akan ditampilkan dalam daftar temuan yang diselesaikan alih-alih daftar temuan aktif.

> **Note**
>
> Ini tidak berlaku untuk temuan Kesalahan. Ketika IAM Access Analyzer tidak dapat menganalisis sumber daya, itu akan menghasilkan temuan kesalahan. Jika Anda mengatasi masalah yang mencegah IAM Access Analyzer menganalisis sumber daya, temuan kesalahan akan dihapus sepenuhnya alih-alih mengubah ke temuan yang diselesaikan.

Jika perubahan yang Anda buat mengakibatkan sumber daya dibagikan di luar zona kepercayaan Anda, tetapi dengan cara yang berbeda, seperti dengan prinsipal yang berbeda atau untuk izin yang berbeda, IAM Access Analyzer akan menghasilkan temuan Aktif baru.

> **Note**
>
> Diperlukan waktu hingga 30 menit setelah kebijakan diubah agar IAM Access Analyzer kembali menganalisis sumber daya dan kemudian memperbarui temuannya. Temuan yang diselesaikan dihapus 90 hari setelah pembaruan terakhir ke status temuan.

Menyelesaikan temuan akses yang tidak terpakai

IAMAccess Analyzer menyediakan langkah-langkah yang direkomendasikan untuk menyelesaikan temuan penganalisis akses yang tidak digunakan berdasarkan jenis temuan.

Setelah Anda membuat perubahan untuk menyelesaikan temuan akses yang tidak digunakan, status temuan diubah menjadi Terselesaikan saat berikutnya penganalisis akses yang tidak

digunakan berjalan. Temuan ini tidak lagi ditampilkan dalam daftar temuan aktif dan sebagai gantinya ditampilkan dalam daftar temuan yang diselesaikan. Jika Anda membuat perubahan yang hanya sebagian membahas temuan akses yang tidak digunakan, temuan yang ada diubah menjadi Terselesaikan tetapi temuan baru dihasilkan. Misalnya, jika Anda menghapus hanya beberapa izin yang tidak digunakan dalam temuan, tetapi tidak semuanya.

IAMAccess Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Menyelesaikan temuan izin yang tidak digunakan

Untuk temuan izin yang tidak digunakan, IAM Access Analyzer dapat merekomendasikan kebijakan untuk dihapus dari IAM pengguna atau peran dan memberikan kebijakan baru untuk mengganti kebijakan izin yang ada. Rekomendasi kebijakan tidak didukung untuk skenario berikut:

- Temuan izin yang tidak digunakan adalah untuk IAM pengguna yang berada dalam grup pengguna.
- Temuan izin yang tidak digunakan adalah untuk IAM peran Pusat IAM Identitas.
- Temuan izin yang tidak digunakan memiliki kebijakan izin yang sudah ada yang menyertakan elemen. `notAction`

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Finding of Unused Permissions.
4. Di bagian Rekomendasi, jika ada kebijakan yang tercantum di kolom Kebijakan yang disarankan, pilih Kebijakan pratinjau untuk melihat kebijakan yang ada dengan kebijakan yang disarankan untuk mengganti kebijakan yang ada. Jika ada beberapa kebijakan yang direkomendasikan, Anda dapat memilih Kebijakan Berikutnya dan Kebijakan Sebelumnya untuk melihat setiap kebijakan yang ada dan yang direkomendasikan.
5. Pilih Unduh JSON untuk mengunduh file.zip dengan JSON file dari semua kebijakan yang disarankan.
6. Buat dan lampirkan kebijakan yang disarankan ke IAM pengguna atau peran. Untuk informasi selengkapnya, lihat Mengubah izin untuk pengguna (konsol) dan Memodifikasi kebijakan izin peran (konsol).

7. Hapus kebijakan yang tercantum dalam kolom Kebijakan izin yang ada dari IAM pengguna atau peran. Untuk informasi selengkapnya, lihat Menghapus izin dari pengguna (konsol) dan Memodifikasi kebijakan izin peran (konsol).

Menyelesaikan temuan peran yang tidak digunakan

Untuk temuan peran yang tidak digunakan, IAM Access Analyzer merekomendasikan untuk menghapus peran yang tidak digunakan. IAM

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Finding of Unused role.
4. Di bagian Rekomendasi, tinjau detail IAM peran.
5. Hapus IAM peran. Untuk informasi selengkapnya, lihat Menghapus IAM peran (konsol).

Menyelesaikan temuan kunci akses yang tidak digunakan

Untuk temuan kunci akses yang tidak digunakan, IAM Access Analyzer merekomendasikan untuk menonaktifkan atau menghapus kunci akses yang tidak digunakan.

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Finding of Unused access keys.
4. Di bagian Rekomendasi, tinjau detail kunci akses.
5. Nonaktifkan atau hapus kunci akses. Untuk informasi selengkapnya, lihat Mengelola kunci akses (konsol).

Menyelesaikan temuan kata sandi yang tidak digunakan

Untuk temuan kata sandi yang tidak digunakan, IAM Access Analyzer merekomendasikan untuk menghapus kata sandi yang tidak digunakan untuk pengguna. IAM

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Menemukan kata sandi yang tidak digunakan.

4. Di bagian Rekomendasi, tinjau detail IAM pengguna.
5. Hapus kata sandi untuk IAM pengguna. Untuk informasi selengkapnya, lihat Membuat, mengubah, atau menghapus kata sandi IAM pengguna (konsol).

IAMAkses jenis sumber daya Analyzer untuk akses eksternal

Untuk penganalisis akses eksternal, IAM Access Analyzer menganalisis kebijakan berbasis sumber daya yang diterapkan ke sumber AWS daya di Wilayah tempat Anda mengaktifkan Access Analyzer. IAM Ini hanya menganalisis kebijakan berbasis sumber daya. Tinjau informasi tentang setiap sumber daya untuk detail tentang bagaimana IAM Access Analyzer menghasilkan temuan untuk setiap jenis sumber daya.

> **Note**
>
> Jenis sumber daya yang didukung yang tercantum adalah untuk penganalisis akses eksternal. Penganalisis akses yang tidak digunakan hanya mendukung IAM pengguna dan peran. Untuk informasi selengkapnya, lihat Memahami cara kerja temuan IAM Access Analyzer.

Jenis sumber daya yang didukung untuk akses eksternal:

- Bucket Amazon Simple Storage Service
- Ember direktori Layanan Penyimpanan Sederhana Amazon
- AWS Identity and Access Management peran
- AWS Key Management Service kunci
- AWS Lambda fungsi dan lapisan
- Antrean Amazon Simple Queue Service
- AWS Secrets Manager rahasia
- Topik Amazon Simple Notification Service
- Cuplikan volume Amazon Elastic Block Store
- Cuplikan DB Layanan Amazon Relational Database Service
- Cuplikan cluster DB Layanan Relational Database Service Amazon
- Repositori Registri Wadah Elastis Amazon
- Sistem file Amazon Elastic File System

- Aliran Amazon DynamoDB
- Tabel Amazon DynamoDB

Bucket Amazon Simple Storage Service

Saat IAM Access Analyzer menganalisis bucket Amazon S3, maka akan menghasilkan temuan saat kebijakan bucket Amazon S3, atau titik aksesACL, termasuk titik akses Multi-wilayah, diterapkan ke bucket memberikan akses ke entitas eksternal. Entitas eksternal adalah penanggung jawab atau entitas lain yang dapat Anda gunakan untuk buat filter yang tidak berada dalam zona kepercayaan Anda. Misalnya, jika kebijakan bucket memberikan akses ke akun lain atau mengizinkan akses publik, IAM Access Analyzer akan menghasilkan temuan. Namun, jika Anda mengaktifkan Blokir Akses Publik di bucket, Anda dapat memblokir akses di tingkat akun atau tingkat bucket.

> **Note**
>
> IAMAccess Analyzer tidak menganalisis kebijakan titik akses yang dilampirkan ke titik akses lintas akun karena titik akses dan kebijakannya berada di luar akun penganalisis. IAMAccess Analyzer menghasilkan temuan publik saat bucket mendelegasikan akses ke titik akses lintas akun dan Blokir Akses Publik tidak diaktifkan di bucket atau akun. Saat Anda mengaktifkan Blokir Akses Publik, temuan publik diselesaikan dan IAM Access Analyzer menghasilkan temuan lintas akun untuk titik akses lintas akun.

Setelan Amazon S3 Blokir Akses Publik mengesampingkan kebijakan bucket yang diterapkan ke bucket. Pengaturan tersebut juga membatalkan kebijakan titik akses yang berlaku pada titik akses bucket. IAMAccess Analyzer menganalisis setelan Blokir Akses Publik di tingkat bucket setiap kali kebijakan berubah. Namun, ini mengevaluasi pengaturan Blokir Akses Publik di tingkat akun hanya sekali setiap 6 jam. Ini berarti IAM Access Analyzer mungkin tidak menghasilkan atau menyelesaikan temuan untuk akses publik ke bucket hingga 6 jam. Misalnya, jika Anda memiliki kebijakan bucket yang memungkinkan akses publik, IAM Access Analyzer akan menghasilkan temuan untuk akses tersebut. Jika Anda kemudian mengaktifkan Blokir Akses Publik untuk memblokir semua akses publik ke bucket di tingkat akun, IAM Access Analyzer tidak menyelesaikan temuan kebijakan bucket hingga 6 jam, meskipun semua akses publik ke bucket diblokir. Resolusi temuan publik untuk jalur akses lintas akun juga dapat memakan waktu hingga 6 jam setelah Anda mengaktifkan Blokir Akses Publik di tingkat akun.

Untuk jalur akses Multi-wilayah, IAM Access Analyzer menggunakan kebijakan yang ditetapkan untuk menghasilkan temuan. IAMAccess Analyzer mengevaluasi perubahan pada titik akses Multi-region setiap 6 jam sekali. Ini berarti IAM Access Analyzer tidak menghasilkan atau menyelesaikan temuan hingga 6 jam, meskipun Anda membuat atau menghapus jalur akses Multi-wilayah, atau memperbarui kebijakan untuk itu.

Ember direktori Layanan Penyimpanan Sederhana Amazon

Bucket direktori Amazon S3 menggunakan kelas penyimpanan Amazon S3 Express One, yang direkomendasikan untuk beban kerja atau aplikasi yang kritis terhadap kinerja. Untuk bucket direktori Amazon S3, IAM Access Analyzer menganalisis kebijakan bucket direktori, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses bucket direktori. Untuk informasi selengkapnya tentang bucket direktori Amazon S3, lihat Bucket direktori di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

AWS Identity and Access Management peran

Untuk IAM peran, IAM Access Analyzer menganalisis kebijakan kepercayaan. Dalam kebijakan kepercayaan peran, Anda menetapkan penanggung jawab yang Anda percayai untuk mengasumsikan peran tersebut. Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya wajib yang melekat pada peran di dalamnya. IAM IAMAccess Analyzer menghasilkan temuan untuk peran dalam zona kepercayaan yang dapat diakses oleh entitas eksternal yang berada di luar zona kepercayaan Anda.

> **Note**
>
> IAMPeran adalah sumber daya global. Jika kebijakan kepercayaan peran memberikan akses ke entitas eksternal, IAM Access Analyzer akan menghasilkan temuan di setiap Wilayah yang diaktifkan.

AWS Key Management Service kunci

Untuk AWS KMS keys, IAM Access Analyzer menganalisis kebijakan utama dan hibah yang diterapkan pada kunci. IAMAccess Analyzer menghasilkan temuan jika kebijakan atau hibah kunci memungkinkan entitas eksternal untuk mengakses kunci. Misalnya, jika Anda menggunakan kunci CallerAccount kondisi kms: dalam pernyataan kebijakan untuk mengizinkan akses ke semua pengguna di AWS akun tertentu, dan Anda menentukan akun selain akun saat ini (zona kepercayaan untuk penganalisis saat ini), IAM Access Analyzer menghasilkan temuan. Untuk mempelajari

selengkapnya tentang kunci AWS KMS kondisi dalam pernyataan IAM kebijakan, lihat Kunci AWS KMS Kondisi.

Ketika IAM Access Analyzer menganalisis kunci, KMS kunci akan membaca metadata kunci, seperti kebijakan kunci dan daftar hibah. Jika kebijakan kunci tidak mengizinkan peran IAM Access Analyzer membaca metadata kunci, pencarian error Access Denied akan dihasilkan. Misalnya, jika pernyataan kebijakan contoh berikut adalah satu-satunya kebijakan yang diterapkan pada kunci, itu menghasilkan pencarian kesalahan Access ditolak di IAM Access Analyzer.

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

Karena pernyataan ini hanya mengizinkan peran bernama Admin dari AWS akun 111122223333 untuk mengakses kunci, pencarian kesalahan Access Denied dihasilkan karena IAM Access Analyzer tidak dapat sepenuhnya menganalisis kunci. Temuan kesalahan ditampilkan dalam teks merah di tabel Temuan. Temuan ini terlihat mirip dengan yang berikut ini.

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

Saat Anda membuat KMS kunci, izin yang diberikan untuk mengakses kunci bergantung pada cara Anda membuat kunci. Jika Anda menerima pencarian kesalahan Access Denied untuk sumber daya kunci, terapkan pernyataan kebijakan berikut ke sumber daya kunci untuk memberikan izin IAM Access Analyzer untuk mengakses kunci.

```
{
```

```
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

Setelah Anda menerima temuan Akses Ditolak untuk sumber daya KMS utama, dan kemudian menyelesaikan temuan dengan memperbarui kebijakan kunci, temuan akan diperbarui ke status Terselesaikan. Jika ada pernyataan kebijakan atau kunci izin yang mengizinkan kunci ke entitas eksternal, Anda mungkin melihat temuan tambahan untuk sumber daya kunci.

AWS Lambda fungsi dan lapisan

Untuk AWS Lambda fungsi, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memberikan akses ke fungsi ke entitas eksternal. Dengan Lambda, Anda dapat melampirkan kebijakan berbasis sumber daya unik ke fungsi, versi, alias, dan lapisan. IAMAccess Analyzer melaporkan akses eksternal berdasarkan kebijakan berbasis sumber daya yang dilampirkan pada fungsi dan lapisan. IAMAccess Analyzer tidak melaporkan akses eksternal berdasarkan kebijakan berbasis sumber daya yang dilampirkan ke alias dan versi tertentu yang dipanggil menggunakan kualifikasi. ARN

Untuk informasi selengkapnya, lihat Menggunakan kebijakan berbasis sumber daya untuk Lambda dan Menggunakan versi di Panduan Pengembang. AWS Lambda

Antrean Amazon Simple Queue Service

Untuk SQS antrian Amazon, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses antrian.

AWS Secrets Manager rahasia

Untuk AWS Secrets Manager rahasia, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses rahasia.

Topik Amazon Simple Notification Service

IAMAccess Analyzer menganalisis kebijakan berbasis sumber daya yang dilampirkan pada topik SNS Amazon, termasuk pernyataan kondisi dalam kebijakan yang memungkinkan akses eksternal ke topik. Anda dapat mengizinkan akun eksternal untuk melakukan SNS tindakan Amazon seperti berlangganan dan menerbitkan topik melalui kebijakan berbasis sumber daya. SNSTopik Amazon dapat diakses secara eksternal jika kepala sekolah dari akun di luar zona kepercayaan Anda dapat melakukan operasi pada topik tersebut. Ketika Anda memilih `Everyone` dalam kebijakan Anda saat membuat SNS topik Amazon, Anda membuat topik dapat diakses oleh publik. `AddPermission`adalah cara lain untuk menambahkan kebijakan berbasis sumber daya ke SNS topik Amazon yang memungkinkan akses eksternal.

Cuplikan volume Amazon Elastic Block Store

Snapshot volume Amazon Elastic Block Store tidak memiliki kebijakan berbasis sumber daya. Snapshot dibagikan melalui izin EBS berbagi Amazon. Untuk snapshot EBS volume Amazon, IAM Access Analyzer menganalisis daftar kontrol akses yang memungkinkan entitas eksternal mengakses snapshot. Snapshot EBS volume Amazon dapat dibagikan dengan akun eksternal saat dienkripsi. Snapshot volume yang tidak terenkripsi dapat dibagikan dengan akun eksternal dan memberikan akses publik. Pengaturan berbagi ada di `CreateVolumePermissions` atribut snapshot. Saat pelanggan melihat pratinjau akses eksternal EBS snapshot Amazon, mereka dapat menentukan kunci enkripsi sebagai indikator bahwa snapshot dienkripsi, mirip dengan cara pratinjau IAM Access Analyzer menangani rahasia Secrets Manager.

Cuplikan DB Layanan Amazon Relational Database Service

Snapshot Amazon RDS DB tidak memiliki kebijakan berbasis sumber daya. Snapshot DB dibagikan melalui izin RDS database Amazon, dan hanya snapshot DB manual yang dapat dibagikan. Untuk snapshot Amazon RDS DB, IAM Access Analyzer menganalisis daftar kontrol akses yang memungkinkan entitas eksternal mengakses snapshot. Snapshot DB yang tidak terenkripsi dapat bersifat publik. Snapshot DB terenkripsi tidak dapat dibagikan secara publik, tetapi dapat dibagikan dengan hingga 20 akun lainnya. Untuk informasi selengkapnya, lihat Membuat snapshot DB. IAMAccess Analyzer menganggap kemampuan untuk mengekspor snapshot manual database (misalnya, ke bucket Amazon S3) sebagai akses tepercaya.

> **Note**
>
> IAMAccess Analyzer tidak mengidentifikasi akses publik atau lintas akun yang dikonfigurasi langsung pada database itu sendiri. IAMAccess Analyzer hanya mengidentifikasi temuan untuk akses publik atau lintas akun yang dikonfigurasi pada snapshot Amazon RDS DB.

Cuplikan cluster DB Layanan Relational Database Service Amazon

Snapshot kluster Amazon RDS DB tidak memiliki kebijakan berbasis sumber daya. Snapshot dibagikan melalui izin cluster Amazon RDS DB. Untuk snapshot kluster Amazon RDS DB, IAM Access Analyzer menganalisis daftar kontrol akses yang memungkinkan entitas eksternal mengakses snapshot. Snapshot cluster yang tidak terenkripsi dapat bersifat publik. Snapshot kluster terenkripsi tidak dapat dibagikan secara publik. Snapshot cluster yang tidak terenkripsi dan terenkripsi dapat dibagikan dengan hingga 20 akun lainnya. Untuk informasi selengkapnya, lihat Membuat snapshot cluster DB. IAMAccess Analyzer menganggap kemampuan untuk mengekspor snapshot cluster DB (misalnya, ke bucket Amazon S3) sebagai akses tepercaya.

> **Note**
>
> IAMTemuan Access Analyzer tidak termasuk pemantauan bagian cluster Amazon RDS DB dan klon dengan yang lain Akun AWS atau organisasi yang menggunakan. AWS Resource Access Manager IAMAccess Analyzer hanya mengidentifikasi temuan untuk akses publik atau lintas akun yang dikonfigurasi pada snapshot klaster Amazon RDS DB.

Repositori Registri Wadah Elastis Amazon

Untuk ECR repositori Amazon, IAM Access Analyzer menganalisis kebijakan berbasis sumber daya, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses repositori (mirip dengan jenis sumber daya lain seperti topik Amazon dan sistem file Amazon). SNS EFS Untuk ECR repositori Amazon, kepala sekolah harus memiliki izin untuk `ecr:GetAuthorizationToken` melalui kebijakan berbasis identitas agar dianggap tersedia secara eksternal.

Sistem file Amazon Elastic File System

Untuk sistem EFS file Amazon, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses ke sistem file. Sistem

EFS file Amazon dapat diakses secara eksternal jika prinsipal dari akun di luar zona kepercayaan Anda dapat melakukan operasi pada sistem file tersebut. Akses didefinisikan oleh kebijakan sistem file yang menggunakanIAM, dan oleh bagaimana sistem file dipasang. Misalnya, memasang sistem EFS file Amazon Anda di akun lain dianggap dapat diakses secara eksternal, kecuali akun tersebut ada di organisasi Anda dan Anda telah mendefinisikan organisasi sebagai zona kepercayaan Anda. Jika Anda memasang sistem file dari cloud pribadi virtual dengan subnet publik, sistem file dapat diakses secara eksternal. Ketika Anda menggunakan Amazon EFS dengan AWS Transfer Family, permintaan akses sistem file yang diterima dari server Transfer Family yang dimiliki oleh akun yang berbeda dari sistem file diblokir jika sistem file memungkinkan akses publik.

Aliran Amazon DynamoDB

IAMAccess Analyzer menghasilkan temuan jika kebijakan DynamoDB mengizinkan setidaknya satu tindakan lintas akun yang memungkinkan entitas eksternal mengakses aliran DynamoDB. Untuk informasi selengkapnya tentang tindakan lintas akun yang didukung untuk DynamoDB, IAMlihat tindakan yang didukung oleh kebijakan berbasis sumber daya di Panduan Pengembang Amazon DynamoDB.

Tabel Amazon DynamoDB

IAMAccess Analyzer menghasilkan temuan untuk tabel DynamoDB jika kebijakan DynamoDB mengizinkan setidaknya satu tindakan lintas akun yang memungkinkan entitas eksternal mengakses tabel atau indeks DynamoDB. Untuk informasi selengkapnya tentang tindakan lintas akun yang didukung untuk DynamoDB, IAMlihat tindakan yang didukung oleh kebijakan berbasis sumber daya di Panduan Pengembang Amazon DynamoDB.

Administrator yang didelegasikan untuk IAM Access Analyzer

Jika Anda mengonfigurasi AWS Identity and Access Management Access Analyzer di akun AWS Organizations manajemen, Anda dapat menambahkan akun anggota di organisasi sebagai administrator yang didelegasikan untuk mengelola IAM Access Analyzer untuk organisasi Anda. Administrator yang didelegasikan memiliki izin untuk membuat dan mengelola penganalisis dalam organisasi. Hanya akun manajemen yang dapat menambahkan administrator yang didelegasikan.

Administrator yang didelegasikan untuk IAM Access Analyzer adalah akun anggota dalam organisasi yang memiliki izin untuk membuat dan mengelola penganalisis yang menganalisis akses di seluruh organisasi. Hanya akun manajemen yang dapat menambahkan, menghapus, atau mengubah administrator yang telah didelegasikan.

Jika Anda menambahkan administrator yang didelegasikan, Anda dapat berganti ke akun lain untuk administrator yang didelegasikan. Ketika Anda melakukannya, akun administrator yang didelegasikan sebelumnya kehilangan izin untuk semua penganalisis yang dibuat menggunakan akun tersebut untuk menganalisis akses di seluruh organisasi. Penganalisis ini berpindah ke status dinonaktifkan dan tidak lagi menghasilkan temuan baru atau memperbarui temuan yang ada. Temuan yang ada untuk penganalisis ini juga tidak lagi dapat diakses. Anda dapat mengaksesnya di masa mendatang dengan mengonfigurasi akun tersebut sebagai administrator yang didelagasikan. Jika Anda tahu bahwa Anda tidak akan menggunakan akun yang sama sebagai administrator yang didelegasikan, pertimbangkan untuk menghapus penganalisis sebelum mengubah administrator yang didelegasikan. Ini akan menghapus semua temuan yang dihasilkan. Ketika administrator baru yang dihilangkan membuat penganalisis baru, maka akan menghasilkan instans baru dari temuan yang sama. Anda tidak kehilangan temuan apa pun, mereka hanya dihasilkan untuk penganalisis baru di akun lain. Dan Anda dapat terus mengakses temuan untuk organisasi menggunakan akun manajemen organisasi, yang juga memiliki izin administrator. Administrator baru yang didelegasikan harus membuat penganalisis baru untuk IAM Access Analyzer untuk mulai memantau sumber daya di organisasi Anda.

Jika administrator yang didelegasikan meninggalkan AWS organisasi, hak istimewa administrasi yang didelegasikan akan dihapus dari akun. Semua penganalisis di akun dengan organisasi sebagai zona kepercayaan berpindah ke keadaan dinonaktifkan. Temuan yang ada untuk penganalisis ini juga tidak lagi dapat diakses.

Saat pertama kali Anda mengonfigurasi penganalisis di akun manajemen, Anda dapat memilih Tambahkan administrator yang didelegasikan di halaman pengaturan Analyzer di konsol IAM Access Analyzer.

> **Note**
>
> IAMAccess Analyzer mengenakan biaya untuk penganalisis akses yang tidak digunakan berdasarkan jumlah IAM peran dan pengguna yang dianalisis per penganalisis per bulan. Jika Anda membuat penganalisis akses yang tidak digunakan di akun manajemen dan akun administrator yang didelegasikan, Anda akan dikenakan biaya untuk kedua penganalisis akses yang tidak digunakan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Setelah Anda mengubah administrator yang didelegasikan, administrator yang baru harus membuat penganalisis untuk mulai memantau akses ke sumber daya di organisasi Anda.

Menambahkan administrator yang didelegasikan untuk IAM Access Analyzer

Jika Anda mengonfigurasi AWS Identity and Access Management Access Analyzer di akun AWS Organizations manajemen, Anda dapat menambahkan akun anggota di organisasi sebagai administrator yang didelegasikan untuk mengelola IAM Access Analyzer untuk organisasi Anda. Administrator yang didelegasikan memiliki izin untuk membuat dan mengelola penganalisis dalam organisasi. Hanya akun manajemen yang dapat menambahkan administrator yang didelegasikan.

Untuk menambahkan administrator yang didelegasikan dengan menggunakan konsol

1. Masuk ke AWS konsol menggunakan akun manajemen untuk organisasi Anda.
2. Buka IAM konsol di https://console.aws.amazon.com/iam/.
3. Di bawah Access Analyzer, pilih Pengaturan Analyzer.
4. Pilih Tambahkan administrator yang didelegasikan.
5. Di bidang Administrator yang didelegasikan, masukkan Akun AWS nomor akun anggota organisasi untuk membuat administrator yang didelegasikan.

   Akun tersebut harus menjadi anggota dari organisasi Anda.
6. Pilih Simpan perubahan.

Untuk menambahkan administrator yang didelegasikan menggunakan AWS CLI atau AWS SDKs

Bila Anda membuat akses analyzer ke analyzer di seluruh organisasi dalam akun administrator yang didelegasikan menggunakan AWS CLI, AWS API (menggunakan AWS SDKs) atau AWS CloudFormation, Anda harus menggunakan AWS Organizations APIs untuk mengaktifkan akses layanan untuk IAM Access Analyzer dan mendaftarkan akun anggota sebagai administrator yang didelegasikan.

1. Aktifkan akses layanan tepercaya untuk IAM Access Analyzer di AWS Organizations. Lihat Cara Mengaktifkan atau Menonaktifkan Akses Tepercaya di Panduan AWS Organizations Pengguna.
2. Daftarkan akun anggota yang valid dari AWS organisasi Anda sebagai administrator yang didelegasikan menggunakan AWS Organizations `RegisterDelegatedAdministrator`APIoperasi atau `register-delegated-administrator` AWS CLI perintah.

Hapus penganalisis akses eksternal dan tidak terpakai

Anda dapat menghapus penganalisis akses eksternal dan tidak terpakai yang ada dari halaman pengaturan Analyzer. Saat Anda menghapus penganalisis, sumber daya yang ditentukan dalam penganalisis tidak lagi dipantau dan tidak ada temuan baru yang dihasilkan. Semua temuan yang dihasilkan oleh penganalisis dihapus.

Untuk temuan yang dihapus karena penganalisis yang menghasilkannya dihapus, acara dikirim ke EventBridge dalam dua hari berikutnya setelah penganalisis dihapus. Diperlukan waktu hingga 90 hari setelah penganalisis dihapus agar temuan Security Hub dihapus.

Untuk menghapus penganalisis

1. Buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di bawah Access Analyzer, pilih Pengaturan Analyzer.
3. Pilih analyzer yang akan dihapus dan kemudian pilih Delete.
4. Ketik **`delete`** kotak teks konfirmasi dan kemudian pilih Hapus.

Aturan arsip

Aturan arsip secara otomatis mengarsipkan temuan baru yang memenuhi kriteria yang Anda tentukan saat membuat aturan. Anda juga dapat menerapkan aturan arsip secara retroaktif untuk mengarsipkan temuan yang ada yang memenuhi kriteria aturan arsip. Misalnya, Anda dapat membuat aturan arsip untuk secara otomatis mengarsipkan temuan apa pun untuk bucket Amazon S3 tertentu yang dapat Anda akses secara teratur. Atau jika Anda memberikan akses ke beberapa sumber daya ke prinsip tertentu, Anda dapat membuat aturan yang secara otomatis mengarsipkan setiap temuan baru yang dihasilkan untuk akses yang diberikan ke prinsip tersebut. Hal ini memungkinkan Anda fokus hanya pada temuan aktif yang mungkin menunjukkan risiko keamanan.

Saat Anda membuat aturan arsip, hanya temuan baru yang cocok dengan kriteria aturan yang diarsipkan secara otomatis. Temuan yang ada tidak diarsipkan secara otomatis. Saat membuat aturan, Anda dapat menyertakan hingga 20 nilai per kriteria di aturan. Untuk daftar kunci filter yang dapat Anda gunakan untuk membuat atau memperbarui aturan arsip, lihat Kunci filter IAM Access Analyzer.

> **Note**
>
> Saat Anda membuat atau mengedit aturan arsip, IAM Access Analyzer tidak memvalidasi nilai yang Anda sertakan dalam filter untuk aturan tersebut. Misalnya, jika Anda menambahkan aturan untuk mencocokkan Akun AWS, IAM Access Analyzer menerima nilai apa pun di bidang, meskipun itu bukan nomor akun yang valid AWS .

Untuk membuat aturan arsip

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Access analyzer, lalu pilih Pengaturan Analyzer.
3. Di bagian Analyzers, pilih analyzer yang ingin Anda buat aturan arsipnya.
4. Pada tab Aturan arsip, pilih Buat aturan arsip.
5. Masukkan nama untuk aturan jika Anda ingin mengubah nama default.
6. Di bagian Aturan, di bawah Kriteria, pilih properti yang sesuai dengan aturan.
7. Pilih kondisi untuk nilai properti, seperti Contains, Is, atau Not Equals.

   Operator yang tersedia tergantung pada properti yang Anda pilih.
8. Secara opsional, tambahkan nilai tambahan untuk properti, atau tambahkan kriteria tambahan untuk aturan. Untuk temuan akses eksternal, untuk memastikan bahwa aturan Anda tidak akan mengarsipkan temuan baru untuk akses publik, Anda juga dapat menyertakan kriteria Akses publik dan mengaturnya ke false.

   Untuk menambahkan nilai lain bagi kriteria, pilih Tambahkan nilai lain. Untuk menambahkan kriteria lain untuk aturan, pilih Tambahkan kriteria.
9. Setelah selesai menambahkan kriteria dan nilai, pilih Buat aturan untuk menerapkan aturan hanya pada temuan baru. Pilih Buat dan arsipkan temuan aktif untuk mengarsipkan temuan baru dan temuan yang ada berdasarkan kriteria aturan. Di bagian Hasil, Anda dapat meninjau daftar temuan aktif yang diterapkan aturan arsip.

Misalnya, untuk membuat aturan untuk temuan akses eksternal yang secara otomatis mengarsipkan temuan apa pun untuk bucket Amazon S3: pilih Jenis sumber daya, lalu pilih Is untuk kondisi tersebut. Selanjutnya pilih bucket S3 dari daftar Nilai.

Untuk membuat aturan untuk temuan akses yang tidak digunakan yang secara otomatis mengarsipkan temuan apa pun untuk akun tertentu: pilih Akun Pemilik Sumber Daya, lalu pilih Sama dengan kondisi tersebut. Ketik Akun AWS ID di kotak teks Nilai.

Lanjutkan untuk menentukan kriteria untuk menyesuaikan aturan yang sesuai untuk lingkungan Anda, lalu pilih Buat aturan.

Jika Anda membuat aturan baru dan menambahkan beberapa kriteria, Anda dapat menghapus satu kriteria dari aturan dengan memilih Hapus kriteria ini. Anda dapat menghapus nilai tambah untuk kriteria dengan memilih Hapus nilai.

Untuk mengedit aturan arsip

1. Pilih nama aturan yang akan diedit di kolom Nama.

   Anda hanya dapat mengedit satu aturan arsip dalam satu waktu.
2. Tambahkan kriteria baru atau hapus kriteria dan nilai yang ada untuk setiap kriteria.
3. Pilih Simpan perubahan untuk menerapkan aturan hanya pada temuan baru. Pilih Simpan dan arsipkan temuan aktif untuk mengarsipkan temuan baru dan temuan yang ada berdasarkan kriteria aturan.

Untuk menghapus aturan arsip

1. Pilih kotak centang untuk aturan yang ingin Anda hapus.
2. Pilih Hapus.
3. Jenis **`delete`** di dialog konfirmasi Hapus aturan arsip, lalu pilih Hapus.

Aturan hanya dihapus dari penganalisis di Wilayah saat ini. Anda harus menghapus aturan arsip secara terpisah untuk setiap penganalisis yang Anda buat di Wilayah lain.

Pemantauan AWS Identity and Access Management Access Analyzer dengan Amazon EventBridge

Gunakan informasi dalam topik ini untuk mempelajari cara memantau temuan IAM Access Analyzer dan mengakses pratinjau dengan Amazon. EventBridge EventBridge adalah versi baru dari Amazon CloudWatch Events.

Peristiwa temuan

IAMAccess Analyzer mengirimkan acara EventBridge untuk setiap temuan yang dihasilkan, untuk perubahan status temuan yang ada, dan saat temuan dihapus. Untuk menerima temuan dan pemberitahuan tentang temuan, Anda harus membuat aturan acara di Amazon EventBridge. Saat Anda membuat aturan peristiwa, Anda juga dapat menetapkan tindakan target yang akan dipicu berdasarkan aturan tersebut. Misalnya, Anda dapat membuat aturan acara yang memicu SNS topik Amazon saat peristiwa untuk temuan baru diterima dari IAM Access Analyzer.

Akses peristiwa pratinjau

IAMAccess Analyzer mengirimkan acara EventBridge untuk setiap pratinjau akses dan mengubah statusnya. Ini termasuk peristiwa ketika pratinjau akses pertama kali dibuat (status Membuat), ketika pratinjau akses selesai (status Selesai), atau ketika pembuatan pratinjau akses gagal (status Gagal). Untuk menerima pemberitahuan tentang pratinjau akses, Anda harus membuat aturan acara di EventBridge. Saat Anda membuat aturan peristiwa, Anda dapat menetapkan tindakan target yang akan dipicu berdasarkan aturan tersebut. Misalnya, Anda dapat membuat aturan peristiwa yang memicu SNS topik Amazon saat peristiwa untuk pratinjau akses selesai diterima dari IAM Access Analyzer.

Frekuensi pemberitahuan peristiwa

IAMAccess Analyzer mengirimkan peristiwa untuk temuan dan temuan baru dengan pembaruan status EventBridge dalam waktu sekitar satu jam sejak peristiwa terjadi di akun Anda. IAMAccess Analyzer juga mengirimkan peristiwa ke EventBridge saat temuan yang diselesaikan dihapus karena periode retensi telah kedaluwarsa. Untuk temuan yang dihapus karena penganalisis yang menghasilkannya dihapus, acara dikirim ke EventBridge sekitar 24 jam setelah penganalisis dihapus. Saat temuan dihapus, status temuan tidak berubah. Sebaliknya, atribut `isDeleted` diatur menjadi `true`. IAMAccess Analyzer juga mengirimkan peristiwa untuk pratinjau akses yang baru dibuat dan perubahan status pratinjau akses ke. EventBridge

Contoh peristiwa temuan akses eksternal

Berikut ini adalah contoh IAM Access Analyzer external access finding event yang dikirim ke EventBridge. Yang `id` tercantum adalah ID untuk acara di EventBridge. Untuk mempelajari lebih lanjut, lihat Peristiwa dan Pola Peristiwa di EventBridge.

Di objek `detail`, nilai untuk atribut `accountId` dan `region` merujuk pada akun dan wilayah yang dilaporkan dalam temuan. Atribut `isDeleted` menunjukkan jika peristiwa berasal dari temuan yang

dihapus. `id` adalah ID temuan. `resources`Array adalah singleton dengan ARN penganalisis yang menghasilkan temuan.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
            "AWS": "999988887777"
        },
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

IAMAccess Analyzer juga mengirimkan peristiwa ke EventBridge untuk temuan kesalahan. Temuan kesalahan adalah temuan yang dihasilkan ketika IAM Access Analyzer tidak dapat menganalisis sumber daya. Peristiwa untuk temuan kesalahan meliputi `error` sebagaimana ditunjukkan dalam contoh berikut.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```


Contoh temuan akses yang tidak digunakan terkait peristiwa

Berikut ini adalah contoh IAM Access Analyzer peristiwa pencarian akses yang tidak digunakan dikirim ke. EventBridge Yang `id` tercantum adalah ID untuk acara di EventBridge. Untuk mempelajari lebih lanjut, lihat Peristiwa dan Pola Peristiwa di EventBridge.

Di objek `detail`, nilai untuk atribut `accountId` dan `region` merujuk pada akun dan wilayah yang dilaporkan dalam temuan. Atribut `isDeleted` menunjukkan jika peristiwa berasal dari temuan yang dihapus. `id` adalah ID temuan.


```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
```

```
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
        }
    }
```

IAMAccess Analyzer juga mengirimkan peristiwa ke EventBridge untuk temuan kesalahan. Temuan kesalahan adalah temuan yang dihasilkan ketika IAM Access Analyzer tidak dapat menganalisis sumber daya. Peristiwa untuk temuan kesalahan meliputi `error` sebagaimana ditunjukkan dalam contoh berikut.

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
```

```
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
  }
```

Contoh peristiwa pratinjau akses

Contoh berikut menunjukkan data untuk acara pertama yang dikirim EventBridge saat Anda membuat pratinjau akses. `resources`Array adalah singleton dengan ARN penganalisis yang dikaitkan dengan pratinjau akses. Di objek `detail`, `id` mengacu pada ID pratinjau akses dan `configuredResources` mengacu pada sumber daya yang pratinjau aksesnya dibuat. `status` adalah `Creating` dan mengacu pada status pratinjau akses. Parameter `previousStatus` tidak ditentukan karena pratinjau akses baru saja dibuat.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
```

```
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Contoh berikut menunjukkan data untuk peristiwa yang dikirim ke EventBridge untuk pratinjau akses dengan perubahan status dari `Creating` ke`Completed`. Dalam objek detail, `id` mengacu pada ID pratinjau akses. `status` dan `previousStatus` mengacu pada status pratinjau akses, ketika status sebelumnya `Creating` dan status saat ini adalah `Completed`.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Contoh berikut menunjukkan data untuk peristiwa yang dikirim ke EventBridge untuk pratinjau akses dengan perubahan status dari `Creating` ke`Failed`. Di objek `detail`, `id` mengacu pada ID pratinjau akses. `status` dan `previousStatus` mengacu pada status pratinjau akses, ketika status sebelumnya `Creating` dan status saat ini adalah `Failed`. Bidang `statusReason` menyediakan kode alasan yang menunjukkan bahwa pratinjau akses gagal karena konfigurasi sumber daya yang tidak valid.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```


Membuat aturan peristiwa menggunakan konsol

Prosedur berikut menjelaskan cara membuat aturan peristiwa menggunakan konsol.

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Dengan menggunakan nilai berikut, buat EventBridge aturan yang memantau pencarian peristiwa atau mengakses peristiwa pratinjau:

   - Untuk Tipe aturan, pilih Aturan dengan pola peristiwa.
   - Untuk sumber acara, pilih Lainnya.
   - Untuk pola Acara, pilih Pola kustom (JSONeditor), dan tempel salah satu contoh pola peristiwa berikut ke area teks:
     - Untuk membuat aturan berdasarkan akses eksternal atau peristiwa temuan akses yang tidak digunakan, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding"
  ]
}
```

- Untuk membuat aturan hanya berdasarkan peristiwa temuan akses yang tidak digunakan, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

> **Note**
>
> Anda tidak dapat membuat aturan hanya berdasarkan peristiwa temuan akses eksternal.

- Untuk membuat aturan berdasarkan peristiwa pratinjau akses, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- Untuk tipe Target, pilih AWS layanan, dan untuk Pilih target, pilih target seperti SNS topik Amazon atau AWS Lambda fungsi. Target terpicu saat peristiwa diterima yang sesuai dengan pola peristiwa yang ditentukan dalam aturan.

Untuk mempelajari selengkapnya tentang cara membuat aturan, lihat Membuat EventBridge aturan Amazon yang bereaksi terhadap peristiwa di Panduan EventBridge Pengguna Amazon.

Membuat aturan acara menggunakan CLI

1. Gunakan yang berikut ini untuk membuat aturan untuk Amazon EventBridge menggunakan AWS CLI. Ganti nama aturan *TestRule* dengan nama untuk aturan Anda.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"]}"
```

2. Anda dapat menyesuaikan aturan untuk memicu tindakan target hanya bagi sebagian temuan yang dihasilkan, seperti temuan dengan atribut spesifik. Contoh berikut ini menunjukkan cara membuat aturan yang memicu tindakan target hanya untuk temuan dengan status Aktif.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":
[\"ACTIVE\"]}}"
```

   Contoh berikut menunjukkan cara membuat aturan yang memicu tindakan target hanya untuk pratinjau akses dengan status from to. `Creating` `Completed`

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. Untuk menetapkan fungsi Lambda sebagai target untuk aturan yang Anda buat, gunakan perintah contoh berikut. Ganti Region dan nama fungsi yang ARN sesuai untuk lingkungan Anda.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. Tambahkan izin yang diperlukan untuk memunculkan target aturan. Contoh berikut menunjukkan cara memberikan izin untuk fungsi Lambda, mengikuti contoh sebelumnya.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Integrasikan IAM Access Analyzer dengan AWS Security Hub

AWS Security Hubmemberikan pandangan komprehensif tentang keadaan keamanan Anda di seluruh AWS. Ini membantu Anda menilai lingkungan Anda terhadap standar industri keamanan dan praktik terbaik. Security Hub mengumpulkan data keamanan dari seluruh Akun AWS, layanan, dan produk mitra pihak ketiga yang didukung. Kemudian menganalisis tren keamanan Anda dan mengidentifikasi masalah keamanan prioritas tertinggi.

Saat mengintegrasikan IAM Access Analyzer dengan Security Hub, Anda dapat mengirim temuan dari IAM Access Analyzer ke Security Hub. Security Hub kemudian dapat memasukkan temuan-temuan tersebut dalam analisisnya tentang postur keamanan Anda secara keseluruhan.

Daftar Isi

Bagaimana IAM Access Analyzer mengirimkan temuan ke Security Hub

Di Security Hub, masalah keamanan dilacak sebagai temuan. Beberapa temuan berasal dari masalah yang terdeteksi oleh orang lain Layanan AWS atau oleh mitra pihak ketiga. Security Hub

juga memiliki seperangkat aturan yang digunakan untuk mendeteksi masalah keamanan dan menghasilkan temuan.

Security Hub menyediakan alat untuk mengelola temuan dari seluruh sumber tersebut. Anda dapat melihat dan memfilter daftar temuan dan melihat informasi terperinci tentang setiap temuan. Untuk informasi lebih lanjut, lihat Melihat temuan di AWS Security Hub Panduan Pengguna. Anda juga dapat melacak status investigasi terhadap temuan. Untuk informasi lebih lanjut, lihat Mengambil tindakan atas temuan di AWS Security Hub Panduan Pengguna.

Semua temuan di Security Hub menggunakan JSON format standar yang disebut AWS Format Pencarian Keamanan (ASFF). ASFFTermasuk rincian tentang sumber masalah, sumber daya yang terpengaruh, dan status temuan saat ini. Untuk informasi selengkapnya, silakan lihat AWS Format Pencarian Keamanan (ASFF) di AWS Security Hub Panduan Pengguna.

AWS Identity and Access Management Access Analyzer adalah salah satu Layanan AWS yang mengirimkan temuan ke Security Hub. Untuk akses yang tidak digunakan, IAM Access Analyzer mendeteksi akses yang tidak terpakai yang diberikan kepada IAM pengguna atau peran dan menghasilkan temuan untuk masing-masing. IAMAccess Analyzer kemudian mengirimkan temuan ini ke Security Hub.

Untuk akses eksternal, IAM Access Analyzer mendeteksi pernyataan kebijakan yang memungkinkan akses publik atau akses lintas akun ke prinsipal eksternal pada sumber daya yang didukung di organisasi atau akun Anda. IAMAccess Analyzer menghasilkan temuan untuk akses publik dan mengirimkannya ke Security Hub. Untuk akses lintas akun, IAM Access Analyzer mengirimkan satu temuan untuk satu prinsipal eksternal sekaligus ke Security Hub. Jika ada beberapa temuan lintas akun di IAM Access Analyzer, Anda harus menyelesaikan temuan Security Hub untuk prinsipal eksternal tunggal sebelum IAM Access Analyzer menyediakan temuan lintas akun berikutnya. Untuk melihat daftar lengkap prinsipal eksternal dengan akses lintas akun di luar zona kepercayaan untuk penganalisis, Anda harus melihat temuan di Access Analyzer. IAM

Jenis temuan yang dikirim IAM Access Analyzer

IAMAccess Analyzer mengirimkan temuan ke Security Hub menggunakan AWS Format Pencarian Keamanan (ASFF). DiASFF, `Types` bidang menyediakan jenis temuan. Temuan dari IAM Access Analyzer dapat memiliki nilai berikut untuk`Types`.

- Temuan akses eksternal - Efek/Paparan Data/Akses Eksternal Diberikan
- Temuan akses eksternal — Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik Keamanan/Akses Eksternal Diberikan

- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik Keamanan/Izin yang Tidak Digunakan
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik IAM Keamanan/Peran yang Tidak Digunakan
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik Keamanan/Kata Sandi Pengguna yang Tidak Digunakan IAM
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik Keamanan/Kunci Akses Pengguna yang Tidak Digunakan IAM

Latensi untuk mengirim temuan

Ketika IAM Access Analyzer membuat temuan baru, biasanya dikirim ke Security Hub dalam waktu 30 menit. Namun, ada kasus yang jarang terjadi ketika IAM Access Analyzer mungkin tidak diberi tahu tentang perubahan kebijakan. Sebagai contoh:

- Perubahan pada pengaturan akses publik blok tingkat akun Amazon S3 dapat memakan waktu hingga 12 jam untuk tercermin dalam Access Analyzer. IAM
- Jika ada masalah pengiriman dengan AWS CloudTrail pengiriman log, perubahan kebijakan mungkin tidak memicu pemindaian ulang sumber daya yang dilaporkan dalam temuan.

Dalam situasi ini, IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui selama pemindaian berkala berikutnya.

Mencoba kembali saat Security Hub tidak tersedia

Jika Security Hub tidak tersedia, IAM Access Analyzer mencoba mengirimkan temuan secara berkala.

Memperbarui temuan yang ada di Security Hub

Setelah mengirim temuan ke Security Hub, IAM Access Analyzer terus mengirim pembaruan untuk mencerminkan pengamatan tambahan dari aktivitas temuan ke Security Hub. Pembaruan ini tercermin dalam temuan yang sama.

Untuk temuan IAM akses eksternal Access Analyzer mengelompokkannya per sumber daya. Di Security Hub, temuan sumber daya tetap aktif jika setidaknya salah satu temuan untuk sumber daya tersebut aktif di IAM Access Analyzer. Jika semua temuan di IAM Access Analyzer untuk sumber

daya diarsipkan atau diselesaikan, maka temuan Security Hub juga diarsipkan. Temuan Security Hub diperbarui saat akses kebijakan berubah antara akses publik dan lintas akun. Pembaruan ini dapat mencakup perubahan jenis, judul, deskripsi, dan tingkat keparahan temuan.

Untuk temuan akses yang tidak digunakan, IAM Access Analyzer tidak mengelompokkannya berdasarkan sumber daya. Sebaliknya, jika temuan akses yang tidak digunakan diselesaikan di IAM Access Analyzer, maka temuan Security Hub terkait juga diselesaikan. Temuan Security Hub diperbarui saat Anda memperbarui IAM pengguna atau peran yang menghasilkan pencarian akses yang tidak digunakan.

Melihat temuan IAM Access Analyzer di Security Hub

Untuk melihat temuan IAM Access Analyzer Anda di Security Hub, pilih Lihat temuan di AWS: IAM Akses bagian Analyzer dari halaman ringkasan. Sebagai alternatif, Anda dapat memilih Temuan dari panel navigasi. Anda kemudian dapat memfilter temuan untuk ditampilkan saja AWS Identity and Access Management Access Analyzer temuan dengan memilih nama Produk: bidang dengan nilai**`IAM Access Analyzer`**.

Menafsirkan IAM Access Analyzer menemukan nama di Security Hub

AWS Identity and Access Management Access Analyzer mengirimkan temuan ke Security Hub menggunakan AWS Format Pencarian Keamanan (ASFF). DiASFF, bidang Jenis menyediakan jenis temuan. ASFFjenis menggunakan skema penamaan yang berbeda dari AWS Identity and Access Management Access Analyzer. Tabel berikut mencakup rincian tentang semua ASFF jenis yang terkait dengan AWS Identity and Access Management Access Analyzer temuan seperti yang muncul di Security Hub.

ASFFmenemukan jenis	Judul pencarian Security Hub	Deskripsi
Efek/Eksposur Data/Akses Eksternal Diberikan	<resource ARN > memungkinkan akses publik	Sebuah kebijakan berbasis sumber daya yang melekat pada sumber daya memungkinkan akses publik pada sumber daya untuk semua penanggung jawab eksternal.
Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik	<resource ARN > memungkinkan akses lintas akun	Kebijakan berbasis sumber daya yang melekat pada

ASFFmenemukan jenis	Judul pencarian Security Hub	Deskripsi
Terbaik Keamanan/Akses Eksternal Diberikan		sumber daya memungkin kan akses lintas akun ke pelaku eksternal di luar zona kepercayaan untuk penganali sis.
Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik Keamanan/Izin yang Tidak Digunakan	<resource ARN > berisi izin yang tidak digunakan	Pengguna atau peran berisi izin layanan dan tindakan yang tidak digunakan.
Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik IAM Keamanan/Peran yang Tidak Digunakan	<resource ARN > berisi peran yang tidak digunakan IAM	Pengguna atau peran berisi peran yang tidak digunakan IAM.
Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik Keamanan/Kata Sandi Pengguna yang Tidak Digunakan IAM	<resource ARN > berisi kata sandi pengguna yang tidak digunakan IAM	Pengguna atau peran berisi kata sandi IAM pengguna yang tidak digunakan.
Pemeriksaan Perangkat Lunak dan Konfigurasi/AWS Praktik Terbaik Keamanan/Kunci Akses Pengguna yang Tidak Digunakan IAM	<resource ARN > berisi kunci akses IAM pengguna yang tidak digunakan	Pengguna atau peran berisi kunci akses IAM pengguna yang tidak digunakan.

Temuan khas dari IAM Access Analyzer

IAMAccess Analyzer mengirimkan temuan ke Security Hub menggunakan AWS Format Pencarian Keamanan (ASFF).

Berikut adalah contoh temuan khas dari IAM Access Analyzer untuk temuan akses eksternal.

```
{
```

```
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::amzn-s3-demo-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-account
 access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-
account access from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
    "Resources": [
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
```

```
    "RecordState": "ACTIVE"
}
```

Berikut adalah contoh temuan khas dari IAM Access Analyzer untuk temuan akses yang tidak digunakan.

```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren’t required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
```

```
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
      "ProductFields": {
      "numberOfUnusedActions": "256",
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
    "Types": [
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```


Mengaktifkan dan mengonfigurasi integrasi

Untuk menggunakan integrasi dengan Security Hub, Anda harus mengaktifkan Security Hub. Untuk informasi tentang cara mengaktifkan Security Hub, lihat Menyiapkan Security Hub di AWS Security Hub Panduan Pengguna.

Saat Anda mengaktifkan IAM Access Analyzer dan Security Hub, integrasi diaktifkan secara otomatis. IAMAccess Analyzer segera mulai mengirim temuan ke Security Hub.

Bagaimana cara menghentikan pengiriman temuan

Untuk menghentikan pengiriman temuan ke Security Hub, Anda dapat menggunakan konsol Security Hub atau konsolAPI.

Lihat Menonaktifkan dan mengaktifkan aliran temuan dari integrasi (konsol) atau Menonaktifkan aliran temuan dari integrasi (Security Hub, API AWS CLI) di AWS Security Hub Panduan Pengguna.

Mencatat panggilan API IAM Access Analyzer dengan AWS CloudTrail

IAM Access Analyzer terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di IAM Access Analyzer. CloudTrail menangkap semua panggilan API untuk IAM Access Analyzer sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari konsol IAM Access Analyzer dan panggilan kode ke operasi API IAM Access Analyzer.

Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail peristiwa secara terus menerus ke bucket Amazon S3, termasuk peristiwa untuk IAM Access Analyzer. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara.

Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk IAM Access Analyzer, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

Informasi IAM Access Analyzer di CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas terjadi di IAM Access Analyzer, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di AWS akun Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk IAM Access Analyzer, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket

Amazon S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua AWS Wilayah. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Semua tindakan IAM Access Analyzer dicatat oleh CloudTrail dan didokumentasikan dalam Referensi API IAM Access Analyzer. Misalnya, panggilan ke`CreateAnalyzer`, `CreateArchiveRule` dan `ListFindings` tindakan menghasilkan entri dalam file CloudTrail log.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut ini:

- Apakah permintaan itu dibuat dengan kredenal pengguna root atau AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi selengkapnya, lihat Elemen CloudTrail UserIdentity.

Memahami entri file log IAM Access Analyzer

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, jadi file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateAnalyzer` operasi yang dilakukan oleh sesi peran yang dianggap bernama “14 Juni `Alice-tempcreds` 2021". Sesi peran dikeluarkan oleh peran bernama`admin-tempcreds`.

```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
  "eventTime": "2021-06-14T22:57:36Z",
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
            "tagkey1": "tagvalue1"
        }
  },
```

```
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Kunci filter IAM Access Analyzer

Anda dapat menggunakan kunci filter di bawah ini untuk menentukan aturan arsip (`CreateArchiveRule`), memperbarui aturan arsip (`UpdateArchiveRule`), mengambil daftar temuan (`ListFindings`dan `ListFindingsV2`), atau mengambil daftar temuan pratinjau akses untuk sumber daya (`ListAccessPreviewFindings`). Tidak ada perbedaan antara menggunakan IAM API dan AWS CloudFormation untuk mengonfigurasi aturan arsip.

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
sumber daya	ARN secara unik mengidentifikasi sumber daya yang dapat diakses oleh penanggung jawab eksternal. Untuk mempelajari lebih lanjut, lihat Nama sumber daya Amazon (ARN).	String	Ya	Ya	Ya
resourceType `AWS::IAM::Role |`	Jenis sumber daya yang dapat diakses oleh penanggung jawab eksternal.	String	Ya	Ya	Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
AWS::KMS::Key | AWS::Lambda::Function | AWS::Lambda::LayerVersion | AWS::S3::Bucket | AWS::S3Express::DirectoryBucket | AWS::SQS::Queue | AWS::SecretsManager::Secret | AWS::EFS::FileSystem | AWS::EC2::Snapshot | AWS::ECR::Repository | AWS::RDS::DBSnapsh					

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
`ot | AWS::RDS::DBClusterSnapshot | AWS::SNS::Topic | AWS::DynamoDB::Stream | AWS::DynamoDB::Table`					
resourceOwnerAccount	12 digit ID AWS akun yang memiliki sumber daya. Untuk mempelajari lebih lanjut, lihat pengidentifikasi akun AWS.	Tali	Ya	Ya	Ya
isPublic	Menunjukkan apakah temuan melaporkan sumber daya yang memiliki kebijakan yang memungkinkan akses publik.	Boolean	Ya	Ya	Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
FindingType `UnusedIAMRole | UnusedIAMUserAccessKey | UnusedIAMUserPassword | UnusedPermission`	Jenis temuannya. Anda hanya dapat memfilter dengan menemukan jenis untuk temuan akses yang tidak digunakan.	String	 Ya	 Ya	 Ya
status `ACTIVE | ARCHIVED | RESOLVED`	Status temuan saat ini.	String	 Tida	 Ya	 Ya
kesalahan	Menunjukkan kesalahan yang dilaporkan untuk temuan.	String	 Ya	 Ya	 Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
Principal .aws	Akun tersebut memberikan akses ke sumber daya di `Principal` bidang temuan. Masukkan ID AWS akun 12 digit atau ARN dari pengguna atau peran AWS eksternal. Untuk mempelajari lebih lanjut, lihat pengidentifikasi akun AWS.	Tali	 Ya	 Ya	 Ya
principal .federated	ARN identitas gabungan yang memiliki akses ke sumber daya di temuan. Untuk mempelajari lebih lanjut, lihat Penyedia identitas dan federasi identitas	String	 Ya	 Ya	 Ya
kondisi.aws: PrincipalArn	ARN penanggung jawab (pengguna, peran, atau kelompok IAM) yang diindikasikan sebagai kondisi untuk akses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	 Ya	 Ya	 Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
condition .aws: ID Principal Org	Pengidentifikasi organisasi penanggung jawab ditunjukkan sebagai kondisi untuk akses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	 Ya	 Ya	 Ya
kondisi.aws: Principal OrgPaths	ID organisasi atau unit organisasi (OU) ditunjukkan sebagai kondisi untuk akses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	 Ya	 Ya	 Ya
kondisi.aws: SourceIp	Alamat IP yang memungkinkan akses penanggung jawab ke sumber daya ketika menggunakan alamat IP yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	Alamat IP	 Ya	 Ya	 Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
kondisi.aws: SourceVpc	ID VPC yang memungkinkan akses penanggung jawab ke sumber daya saat menggunakan VPC yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	 Ya	 Ya	 Ya
kondisi.aws: UserId	ID pengguna pengguna IAM dari akun eksternal yang ditunjukkan sebagai kondisi untuk akses ke sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	 Ya	 Ya	 Ya
condition.cognito-identity.amazonaws.com:id	ID pool identitas Amazon Cognito ditentukan sebagai kondisi untuk akses peran IAM di temuan. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.	String	 Ya	 Ya	 Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
condition.graph.facebook.com:app_id	ID aplikasi Facebook (atau ID situs) yang ditentukan sebagai kondisi untuk memungkinkan Masuk dengan akses federasi Facebook ke peran IAM di temuan. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.	String	 Ya	 Ya	 Ya
condition.accounts.google.com:id	ID aplikasi Google yang ditetapkan sebagai kondisi untuk mengakses peran IAM. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.	String	 Ya	 Ya	 Ya
kondisi.kms:CallerAccount	ID AWS akun yang memiliki entitas pemanggil (pengguna IAM, peran atau pengguna root akun) yang digunakan oleh panggilan layanan. AWS KMS Untuk mempelajari lebih lanjut, lihat Kunci kondisi untuk AWS Key Management Service.	String	 Ya	 Ya	 Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
kondisi.www.amazon.com:app_id	ID aplikasi Amazon (atau ID situs) yang ditentukan sebagai kondisi untuk memungkinkan akses federasi Login with Amazon ke peran. Untuk mempelajari lebih lanjut, lihat	String	 Ya	 Ya	 Ya
id	ID temuan.	String	 Tida	 Ya	 Ya
ChangeType	Menyediakan konteks tentang bagaimana temuan pratinjau akses dibandingkan dengan akses yang ada yang diidentifikasi dalam IAM Access Analyzer.	String	 Tida	 Tida	 Ya
existingFindingId	ID temuan yang ada di IAM Access Analyzer, disediakan hanya untuk temuan yang ada di pratinjau akses.	String	 Tida	 Tida	 Ya
existingFindingStatus	Status yang ada dari temuan, disediakan hanya untuk temuan yang ada di pratinjau akses.	String	 Tida	 Tida	 Ya

Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer menggunakan peran terkait layanan IAM. Peran terkait layanan adalah jenis unik peran IAM yang ditautkan langsung ke IAM Access Analyzer. Peran terkait layanan telah ditentukan sebelumnya oleh IAM Access Analyzer dan mencakup semua izin yang diperlukan fitur untuk memanggil layanan lain atas nama Anda. AWS

Peran terkait layanan membuat pengaturan IAM Access Analyzer lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. IAM Access Analyzer mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya IAM Access Analyzer yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat Layanan AWS yang Berfungsi dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait Layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Izin peran terkait layanan untuk AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer menggunakan peran terkait layanan bernama AWSServiceRoleForAccessAnalyzer— Izinkan Penganalisis Akses untuk menganalisis metadata sumber daya untuk akses eksternal dan menganalisis aktivitas guna mengidentifikasi akses yang tidak digunakan.

Peran AWSServiceRoleForAccessAnalyzer terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `access-analyzer.amazonaws.com`

Kebijakan izin peran bernama `AccessAnalyzerServiceRolePolicy`memungkinkan IAM Access Analyzer menyelesaikan tindakan pada sumber daya tertentu.

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, silakan lihat Izin Peran Tertaut Layanan di Panduan Pengguna IAM.

Membuat peran terkait layanan untuk IAM Access Analyzer

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda mengaktifkan Access Analyzer di AWS Management Console atau AWS API, IAM Access Analyzer akan membuat peran terkait layanan untuk Anda. Peran terkait layanan yang sama digunakan di semua Wilayah tempat Anda mengaktifkan IAM Access Analyzer. Akses eksternal dan temuan akses yang tidak digunakan menggunakan peran terkait layanan yang sama.

> **Note**
>
> IAM Access Analyzer adalah Regional. Anda harus mengaktifkan IAM Access Analyzer di setiap Wilayah secara independen.

Jika Anda menghapus peran terkait layanan ini, IAM Access Analyzer akan membuat ulang peran tersebut saat Anda membuat penganalisis selanjutnya.

Anda juga dapat menggunakan konsol IAM untuk membuat peran yang terhubung dengan layanan dengan kasus penggunaan Penganalisis Akses. Di AWS CLI atau AWS API, buat peran terkait layanan dengan nama `access-analyzer.amazonaws.com` layanan. Untuk informasi lebih lanjut, lihat Membuat Peran yang Terhubung dengan Layanan di Panduan Pengguna IAM. Jika Anda menghapus peran tertaut layanan ini, Anda dapat mengulang proses yang sama untuk membuat peran tersebut lagi.

Mengedit peran terkait layanan untuk IAM Access Analyzer

IAM Access Analyzer tidak mengizinkan Anda mengedit peran terkait AWSServiceRoleForAccessAnalyzer layanan. Setelah Anda membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit Peran Tertaut Layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk IAM Access Analyzer

Jika Anda tidak lagi memerlukan penggunaan fitur atau layanan yang memerlukan peran terkait layanan, kami menyarankan Anda untuk menghapus peran tersebut. Dengan begitu Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus membersihkan sumber daya peran terkait layanan sebelum menghapusnya secara manual.

> **Note**
>
> Jika IAM Access Analyzer menggunakan peran saat Anda mencoba menghapus sumber daya, maka penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus sumber daya IAM Access Analyzer yang digunakan oleh AWSServiceRoleForAccessAnalyzer

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bagian Laporan akses, di bawah Penganalisis akses, pilih Penganalisis.
3. Pilih kotak centang di sebelah kiri atas daftar penganalisis di tabel Penganalisis untuk memilih semua penganalisis.
4. Pilih Hapus.
5. Untuk mengonfirmasi bahwa Anda ingin menghapus penganalisis, masukkan **`delete`**, lalu pilih Hapus.

Untuk menghapus peran terkait layanan secara manual menggunakan IAM

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran AWSServiceRoleForAccessAnalyzer terkait layanan. Untuk informasi selengkapnya, silakan lihat Menghapus Peran Terkait Layanan di Panduan Pengguna IAM.

Wilayah yang Didukung untuk peran terkait layanan IAM Access Analyzer

IAM Access Analyzer mendukung penggunaan peran terkait layanan di semua Wilayah tempat layanan tersedia. Untuk informasi lebih lanjut, lihat Wilayah dan Titik Akhir AWS.

Akses pratinjau

Selain membantu mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal, AWS IAM Access Analyzer juga menampilkan pratinjau temuan IAM Access Analyzer sebelum menerapkan izin sumber daya sehingga Anda dapat memvalidasi bahwa perubahan kebijakan hanya memberikan akses publik dan lintas akun yang dimaksudkan ke sumber daya Anda. Ini membantu Anda memulai dengan akses eksternal yang dimaksudkan ke sumber daya Anda.

Anda dapat melihat pratinjau dan memvalidasi akses publik dan lintas akun ke bucket Amazon S3 Anda di konsol Amazon S3. Anda juga dapat menggunakan IAM Access Analyzer APIs untuk melihat pratinjau akses publik dan lintas akun untuk bucket Amazon S3, kunci, IAM peran, SQS antrian Amazon AWS KMS , dan rahasia Secrets Manager dengan memberikan izin yang diusulkan untuk sumber daya Anda.

Topik

- Mempratinjau akses di konsol Amazon S3
- Mempratinjau akses dengan API IAM Access Analyzer

Mempratinjau akses di konsol Amazon S3

Setelah menyelesaikan kebijakan bucket di konsol Amazon S3, Anda memiliki opsi untuk melihat pratinjau akses publik dan lintas akun ke bucket Amazon S3 Anda. Anda dapat memvalidasi bahwa perubahan kebijakan hanya memberikan akses eksternal yang diinginkan sebelum Anda memilih Simpan perubahan. Langkah opsional ini memungkinkan Anda untuk melihat pratinjau AWS Identity and Access Management Access Analyzer temuan untuk bucket Anda. Anda dapat memvalidasi apakah perubahan kebijakan memperkenalkan temuan baru atau menyelesaikan temuan yang ada untuk akses eksternal. Anda dapat melewati langkah validasi ini dan menyimpan kebijakan bucket Amazon S3 kapan saja.

Untuk melakukan pratinjau akses eksternal ke bucket, Anda harus memiliki penganalisis akun aktif di wilayah bucket dengan akun sebagai zona kepercayaan. Anda juga harus memiliki izin yang diperlukan untuk menggunakan IAM Access Analyzer dan akses pratinjau. Untuk informasi selengkapnya tentang mengaktifkan IAM Access Analyzer dan izin yang diperlukan, lihat. Mengaktifkan IAM Access Analyzer

Untuk melihat pratinjau akses ke bucket Amazon S3 saat Anda membuat atau mengedit kebijakan bucket

1. Setelah selesai membuat atau mengedit kebijakan bucket, pastikan kebijakan Anda adalah kebijakan bucket Amazon S3 yang valid. Kebijakan ARN harus sesuai dengan bucket ARN dan elemen kebijakan harus valid.
2. Di bawah kebijakan, di bagian Akses eksternal pratinjau, pilih penganalisis akun aktif, lalu pilih Pratinjau. Pratinjau temuan IAM Access Analyzer dihasilkan untuk bucket Anda. Pratinjau menganalisis kebijakan bucket Amazon S3 yang ditampilkan, bersama dengan izin bucket yang ada. Ini termasuk pengaturan BPA bucket dan akun, bucket ACL, jalur akses Amazon S3, dan

titik akses multi-wilayah yang terpasang pada bucket, serta kebijakan serta pengaturan BPA mereka.

3. Ketika pratinjau akses selesai, pratinjau temuan IAM Access Analyzer ditampilkan. Setiap temuan melaporkan instance prinsipal di luar akun dengan akses ke bucket Anda setelah Anda menyimpan polis. Anda dapat memvalidasi akses ke bucket Anda dengan meninjau setiap temuan. Header temuan menyediakan ringkasan akses dan Anda dapat memperluas temuan untuk meninjau detail temuan. Badge temuan memberikan konteks tentang bagaimana menyimpan kebijakan bucket akan mengubah akses ke bucket. Misalnya, mereka membantu Anda memvalidasi apakah perubahan kebijakan memperkenalkan temuan baru atau menyelesaikan temuan yang ada untuk akses eksternal:

   a. Baru — menunjukkan temuan untuk akses eksternal baru yang akan diperkenalkan oleh kebijakan tersebut.

   b. Terselesaikan — menunjukkan temuan untuk akses eksternal yang sudah ada yang akan dihapus oleh kebijakan tersebut.

   c. Diarsipkan — menunjukkan temuan untuk akses eksternal baru yang akan diarsipkan secara otomatis, berdasarkan aturan arsip untuk penganalisis yang menentukan kapan temuan harus ditandai sebagaimana dimaksud.

   d. Existing — menunjukkan temuan yang ada untuk akses eksternal yang akan tetap tidak berubah.

   e. Publik — jika sebuah temuan adalah untuk akses publik ke sumber daya, itu akan memiliki lencana Publik, di samping salah satu lencana di atas.

4. Jika Anda mengidentifikasi akses eksternal yang tidak ingin Anda perkenalkan atau hapus, Anda dapat merevisi kebijakan dan kemudian memilih Pratinjau lagi sampai Anda telah mencapai akses eksternal yang Anda inginkan. Jika Anda memiliki temuan berlabel Publik, kami sarankan Anda merevisi kebijakan untuk menghapus akses publik sebelum Anda memilih Simpan perubahan. Pratinjau akses adalah langkah opsional dan Anda dapat memilih Simpan perubahan kapan saja.

Mempratinjau akses dengan API IAM Access Analyzer

Anda dapat menggunakan API IAM Access Analyzer untuk melihat pratinjau akses publik dan lintas akun untuk bucket Amazon S3, AWS KMS kunci, peran IAM, antrian Amazon SQS, dan rahasia Secrets Manager. Anda dapat melihat pratinjau akses dengan memberikan izin yang diajukan untuk sumber daya yang ada yang Anda miliki atau sumber daya baru yang ingin Anda deploy.

Untuk melihat akses eksternal ke sumber daya Anda, Anda harus memiliki penganalisis akun aktif untuk akun dan wilayah sumber daya. Anda juga harus memiliki izin yang diperlukan untuk menggunakan IAM Access Analyzer dan akses pratinjau. Untuk informasi selengkapnya tentang mengaktifkan IAM Access Analyzer dan izin yang diperlukan, lihat. Mengaktifkan IAM Access Analyzer

Untuk melihat pratinjau akses untuk sumber daya, Anda dapat menggunakan operasi `CreateAccessPreview` dan menyediakan penganalisis ARN dan konfigurasi kontrol akses untuk sumber daya. Layanan mengembalikan ID unik untuk pratinjau akses, yang dapat Anda gunakan untuk memeriksa status pratinjau akses dengan operasi `GetAccessPreview`. Ketika statusnya adalah `Completed`, Anda dapat menggunakan operasi `ListAccessPreviewFindings` untuk mengambil temuan yang dihasilkan untuk pratinjau akses. Operasi `GetAccessPreview` dan `ListAccessPreviewFindings` akan mengambil pratinjau akses dan temuan yang dibuat dalam waktu sekitar 24 jam.

Setiap temuan yang diambil berisi detail temuan yang menggambarkan akses. Status pratinjau dari temuan yang menjelaskan apakah temuan akan berupa `Active`, `Archived`, atau `Resolved` setelah deployment izin, dan `changeType`. Konteks `changeType` menyediakan tentang bagaimana temuan pratinjau akses dibandingkan dengan akses yang ada yang diidentifikasi dalam IAM Access Analyzer:

- Baru — temuannya adalah untuk akses yang baru diperkenalkan.
- Tidak berubah — temuan pratinjau adalah temuan yang ada yang akan tetap tidak berubah.
- Berubah - temuan pratinjau adalah temuan yang ada dengan perubahan status.

`status` dan `changeType` membantu Anda memahami bagaimana konfigurasi sumber daya akan mengubah akses sumber daya yang ada. Jika `changeType` ada `Unchanged` atau Diubah, temuan juga akan berisi ID dan status temuan yang ada di IAM Access Analyzer. Misalnya, temuan `Changed` dengan status pratinjau `Resolved` dan status saat ini `Active` mengindikasikan temuan `Active` yang ada untuk sumber daya akan menjadi `Resolved` sebagai hasil dari perubahan izin yang diajukan.

Anda dapat menggunakan operasi `ListAccessPreviews` untuk memperoleh daftar pratinjau akses untuk penganalisis tertentu. Operasi ini akan mengambil informasi tentang pratinjau akses yang dibuat dalam waktu sekitar satu jam.

Secara umum, jika pratinjau akses adalah untuk sumber daya yang ada dan Anda meninggalkan opsi konfigurasi yang tidak ditentukan, pratinjau akses akan menggunakan konfigurasi sumber daya yang

ada secara default. Jika pratinjau akses adalah untuk sumber daya baru dan Anda meninggalkan opsi konfigurasi yang tidak ditentukan, pratinjau akses akan menggunakan nilai default tergantung jenis sumber daya. Untuk kasus konfigurasi untuk setiap jenis sumber daya, lihat di bawah ini.

Pratinjau akses ke bucket Amazon S3 Anda

Untuk membuat pratinjau akses untuk bucket Amazon S3 baru atau bucket Amazon S3 yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi bucket dengan menentukan kebijakan bucket Amazon S3, bucket ACL, pengaturan bucket BPA, dan jalur akses Amazon S3, termasuk titik akses multi-wilayah, yang dilampirkan ke bucket.

> **Note**
>
> Sebelum mencoba membuat pratinjau akses untuk bucket baru, kami sarankan Anda memanggil operasi Amazon HeadBucketS3 untuk memeriksa apakah bucket bernama sudah ada. Operasi ini berguna untuk menentukan apakah bucket ada dan Anda memiliki izin untuk mengaksesnya.

Kebijakan Bucket — Jika konfigurasi untuk bucket Amazon S3 yang sudah ada dan Anda tidak menentukan kebijakan bucket Amazon S3, pratinjau akses menggunakan kebijakan yang ada yang dilampirkan ke bucket. Jika pratinjau akses untuk sumber daya baru dan Anda tidak menentukan kebijakan bucket Amazon S3, pratinjau akses akan mengasumsikan bucket tanpa kebijakan. Untuk mengusulkan penghapusan kebijakan bucket yang ada, Anda dapat menentukan string kosong. Untuk informasi selengkapnya tentang kebijakan bucket yang didukung, lihat Contoh kebijakan bucket.

Hibah ACL Bucket — Anda dapat mengusulkan hingga 100 hibah ACL per ember. Jika konfigurasi izin diusulkan untuk bucket yang ada, pratinjau akses menggunakan daftar diusulkan konfigurasi izin di tempat izin yang ada. Jika tidak, pratinjau akses menggunakan izin yang ada untuk bucket.

Titik akses bucket — Analisis ini mendukung hingga 100 titik akses, termasuk titik akses multi-wilayah, per ember, termasuk hingga sepuluh titik akses baru yang dapat Anda usulkan per ember. Jika konfigurasi titik akses Amazon S3 yang diusulkan adalah untuk bucket yang ada, pratinjau akses menggunakan konfigurasi titik akses yang diusulkan sebagai pengganti titik akses yang ada. Untuk mengusulkan jalur akses tanpa kebijakan, Anda dapat memberikan string kosong sebagai kebijakan titik akses. Untuk informasi selengkapnya tentang batas kebijakan titik akses, lihat Pembatasan dan batasan titik akses.

Blokir konfigurasi akses publik — Jika konfigurasi yang diusulkan adalah untuk bucket Amazon S3 yang ada dan Anda tidak menentukan konfigurasi, pratinjau akses menggunakan setelan yang ada. Jika konfigurasi yang diusulkan adalah untuk bucket baru dan Anda tidak menentukan konfigurasi bucket BPA, pratinjau akses akan digunakan`false`. Jika konfigurasi yang diusulkan adalah untuk titik akses baru atau titik akses multi-wilayah, dan Anda tidak menentukan konfigurasi BPA titik akses, pratinjau akses menggunakan. `true`

Pratinjau akses ke kunci AWS KMS

Untuk membuat pratinjau akses untuk AWS KMS kunci baru atau AWS KMS kunci yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi AWS KMS kunci dengan menentukan kebijakan kunci dan konfigurasi AWS KMS hibah.

AWS KMS kebijakan kunci — Jika konfigurasi untuk kunci yang ada dan Anda tidak menentukan kebijakan kunci, pratinjau akses menggunakan kebijakan yang ada untuk kunci tersebut. Jika pratinjau akses adalah untuk sumber daya baru dan Anda tidak menentukan kebijakan kunci, maka pratinjau akses menggunakan kebijakan kunci default. Kebijakan kunci yang diusulkan tidak boleh string kosong.

AWS KMS hibah — Analisis ini mendukung hibah hingga 100 KMS per konfigurasi*.* Jika konfigurasi hibah yang diusulkan adalah untuk kunci yang ada, pratinjau akses menggunakan daftar konfigurasi hibah yang diusulkan sebagai pengganti hibah yang ada. Jika tidak, pratinjau akses menggunakan izin yang ada untuk kunci.

Pratinjau akses ke peran IAM Anda

Untuk membuat pratinjau akses untuk peran IAM baru atau peran IAM yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi peran IAM dengan menentukan kebijakan kepercayaan.

Kebijakan kepercayaan peran — Jika konfigurasi untuk peran IAM baru, Anda harus menentukan kebijakan kepercayaan. Jika konfigurasi untuk peran IAM yang sudah ada yang Anda miliki dan Anda tidak mengusulkan kebijakan kepercayaan, pratinjau akses menggunakan kebijakan kepercayaan yang ada untuk peran tersebut. Kebijakan kepercayaan yang diusulkan tidak boleh string kosong.

Pratinjau akses ke antrean Amazon SQS

Untuk membuat pratinjau akses untuk antrean Amazon SQS baru atau antrean Amazon SQS yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi antrian Amazon SQS dengan menentukan kebijakan Amazon SQS untuk antrean.

Kebijakan antrian Amazon SQS — Jika konfigurasi untuk antrean Amazon SQS yang ada dan Anda tidak menentukan kebijakan Amazon SQS, pratinjau akses menggunakan kebijakan Amazon SQS yang ada untuk antrean. Jika pratinjau akses untuk sumber daya baru dan Anda tidak menentukan kebijakan, pratinjau akses akan mengasumsikan antrean Amazon SQS tanpa kebijakan. Untuk mengusulkan penghapusan kebijakan antrian Amazon SQS yang ada, Anda dapat menentukan string kosong untuk kebijakan Amazon SQS.

Pratinjau akses ke rahasia Secrets Manager

Untuk membuat pratinjau akses untuk rahasia Secrets Manager baru atau rahasia Secrets Manager yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi rahasia Secrets Manager dengan menentukan kebijakan rahasia dan kunci AWS KMS enkripsi opsional.

Kebijakan rahasia — Jika konfigurasi untuk rahasia yang ada dan Anda tidak menentukan kebijakan rahasia, pratinjau akses menggunakan kebijakan yang ada untuk rahasia tersebut. Jika pratinjau akses adalah untuk sumber daya baru dan Anda tidak menentukan kebijakan, pratinjau akses mengasumsikan rahasia tanpa kebijakan. Untuk mengusulkan penghapusan kebijakan yang ada, Anda dapat menentukan string kosong.

AWS KMS kunci enkripsi — Jika konfigurasi yang diusulkan adalah untuk rahasia baru dan Anda tidak menentukan ID AWS KMS kunci, pratinjau akses menggunakan kunci KMS default AWS akun. Jika Anda menentukan string kosong untuk ID AWS KMS kunci, pratinjau akses menggunakan kunci KMS default AWS akun.

Cek untuk memvalidasi kebijakan

IAMAccess Analyzer menyediakan pemeriksaan kebijakan yang membantu memvalidasi IAM kebijakan Anda sebelum Anda melampirkannya ke entitas. Ini termasuk pemeriksaan kebijakan dasar yang disediakan oleh validasi kebijakan untuk memvalidasi kebijakan Anda terhadap tata bahasa kebijakan dan AWS praktik terbaik. Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan Anda.

Anda dapat menggunakan pemeriksaan kebijakan khusus untuk memeriksa akses baru berdasarkan standar keamanan Anda. Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Cara kerja pemeriksaan kebijakan khusus

Anda dapat memvalidasi kebijakan Anda terhadap standar keamanan yang ditentukan menggunakan pemeriksaan kebijakan AWS Identity and Access Management Access Analyzer khusus. Anda dapat menjalankan jenis pemeriksaan kebijakan kustom berikut:

- Memeriksa kebijakan referensi: Saat mengedit kebijakan, Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan kebijakan referensi, seperti versi kebijakan yang ada. Anda dapat menjalankan pemeriksaan ini saat mengedit kebijakan menggunakan AWS Command Line Interface (AWS CLI), IAM Access Analyzer API (API), atau editor JSON kebijakan di IAM konsol.

  > **Note**
  >
  > IAMPemeriksaan kebijakan kustom Access Analyzer memungkinkan wildcard dalam `Principal` elemen untuk kebijakan sumber daya referensi.

- Periksa daftar IAM tindakan atau sumber daya: Anda dapat memeriksa untuk memastikan bahwa IAM tindakan atau sumber daya tertentu tidak diizinkan oleh kebijakan Anda. Jika hanya tindakan yang ditentukan, IAM Access Analyzer memeriksa akses tindakan pada semua sumber daya dalam kebijakan. Jika hanya sumber daya yang ditentukan, maka IAM Access Analyzer memeriksa tindakan mana yang memiliki akses ke sumber daya yang ditentukan. Jika kedua tindakan dan sumber daya ditentukan, maka IAM Access Analyzer memeriksa tindakan mana yang ditentukan memiliki akses ke sumber daya yang ditentukan. Anda dapat menjalankan pemeriksaan ini ketika Anda membuat atau mengedit kebijakan menggunakan AWS CLI atauAPI.
- Memeriksa akses publik: Anda dapat memeriksa apakah kebijakan sumber daya dapat memberikan akses publik ke jenis sumber daya tertentu. Anda dapat menjalankan pemeriksaan ini ketika Anda membuat atau mengedit kebijakan menggunakan AWS CLI atauAPI. Jenis pemeriksaan kebijakan kustom ini berbeda dengan akses pratinjau karena pemeriksaan tidak memerlukan konteks penganalisis akses akun atau eksternal apa pun. Pratinjau akses memungkinkan Anda untuk melihat pratinjau temuan IAM Access Analyzer sebelum menerapkan izin sumber daya, sementara pemeriksaan kustom menentukan apakah akses publik mungkin diberikan oleh kebijakan.

Biaya terkait dengan setiap pemeriksaan kebijakan khusus. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Anda dapat menjalankan pemeriksaan kebijakan khusus pada identitas dan kebijakan berbasis sumber daya. Pemeriksaan kebijakan khusus tidak bergantung pada teknik pencocokan pola atau memeriksa log akses untuk menentukan apakah akses baru atau yang ditentukan diizinkan oleh kebijakan. Mirip dengan temuan akses eksternal, pemeriksaan kebijakan khusus dibangun di Zelkova. Zelkova menerjemahkan IAM kebijakan ke dalam pernyataan logis yang setara, dan menjalankan serangkaian pemecah logis tujuan umum dan khusus (teori modulo kepuasan) terhadap masalah. Untuk memeriksa akses baru atau yang ditentukan, IAM Access Analyzer menerapkan Zelkova berulang kali ke kebijakan. Pertanyaan menjadi semakin spesifik untuk mengkarakterisasi kelas-kelas perilaku yang diizinkan oleh kebijakan berdasarkan konten kebijakan. Untuk informasi lebih lanjut tentang teori modulo kepuasan, lihat Teori Modulo Kepuasan.

Dalam kasus yang jarang terjadi, IAM Access Analyzer tidak dapat sepenuhnya menentukan apakah pernyataan kebijakan memberikan akses baru atau yang ditentukan. Dalam kasus tersebut, kesalahan di sisi mendeklarasikan positif palsu dengan gagal pemeriksaan kebijakan khusus. IAMAccess Analyzer dirancang untuk memberikan evaluasi kebijakan yang komprehensif dan berusaha untuk meminimalkan negatif palsu. Pendekatan ini berarti bahwa IAM Access Analyzer memberikan jaminan tingkat tinggi bahwa pemeriksaan yang lulus berarti akses tidak diberikan oleh kebijakan. Anda dapat memeriksa pemeriksaan yang gagal secara manual dengan meninjau pernyataan kebijakan yang dilaporkan dalam respons dari IAM Access Analyzer.

Contoh kebijakan referensi untuk memeriksa akses baru

Anda dapat menemukan contoh untuk kebijakan referensi dan mempelajari cara menyiapkan dan menjalankan pemeriksaan kebijakan kustom untuk akses baru di repositori sampel pemeriksaan kebijakan kustom IAM Access Analyzer. GitHub

Sebelum menggunakan contoh-contoh ini

Sebelum Anda menggunakan contoh kebijakan referensi ini, lakukan hal berikut:

- Tinjau dan sesuaikan kebijakan referensi dengan cermat untuk kebutuhan unik Anda.
- Uji kebijakan referensi di lingkungan Anda secara menyeluruh dengan Layanan AWS yang Anda gunakan.

  Kebijakan referensi menunjukkan implementasi dan penggunaan pemeriksaan kebijakan khusus. Contoh-contoh tersebut tidak dimaksudkan untuk ditafsirkan sebagai rekomendasi AWS resmi atau praktik terbaik untuk diimplementasikan persis seperti yang ditunjukkan. Merupakan tanggung jawab Anda untuk secara hati-hati menguji kebijakan referensi untuk kesesuaiannya untuk menyelesaikan persyaratan keamanan untuk lingkungan Anda.

- Pemeriksaan kebijakan khusus bersifat agnostik lingkungan dalam analisisnya. Analisis mereka hanya mempertimbangkan informasi yang terkandung dalam kebijakan input. Misalnya, pemeriksaan kebijakan khusus tidak dapat memeriksa apakah akun adalah anggota AWS organisasi tertentu. Oleh karena itu, pemeriksaan kebijakan kustom tidak dapat membandingkan akses baru berdasarkan nilai kunci kondisi untuk kunci `aws:PrincipalOrgId`dan `aws:PrincipalAccount`kondisi.

Periksa pemeriksaan kebijakan kustom yang gagal

Jika pemeriksaan kebijakan kustom gagal, respons dari IAM Access Analyzer menyertakan ID pernyataan (`Sid`) pernyataan kebijakan yang menyebabkan pemeriksaan gagal. Meskipun ID pernyataan adalah elemen kebijakan opsional, sebaiknya Anda menambahkan ID pernyataan untuk setiap pernyataan kebijakan. Pemeriksaan kebijakan kustom juga menampilkan indeks pernyataan untuk membantu mengidentifikasi alasan kegagalan pemeriksaan. Indeks pernyataan mengikuti penomoran berbasis nol, di mana pernyataan pertama direferensikan sebagai 0. Ketika ada beberapa pernyataan yang menyebabkan cek gagal, cek hanya mengembalikan satu ID pernyataan pada satu waktu. Kami menyarankan Anda memperbaiki pernyataan yang disorot dalam alasan dan menjalankan kembali cek sampai lulus.

Validasi kebijakan dengan IAM Access Analyzer

Anda dapat memvalidasi kebijakan Anda menggunakan validasi AWS Identity and Access Management Access Analyzer kebijakan. Anda dapat membuat atau mengedit kebijakan menggunakan AWS CLI, AWS API, atau editor JSON kebijakan di IAM konsol. IAMAccess Analyzer memvalidasi kebijakan Anda terhadap tata bahasa IAM kebijakan dan AWS praktik terbaik. Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan Anda. Temuan ini memberikan rekomendasi yang dapat ditindaklanjuti yang membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik keamanan. Untuk melihat daftar pemeriksaan kebijakan dasar yang dijalankan oleh IAM Access Analyzer, lihatReferensi pemeriksaan kebijakan Access Analyzer.

Memvalidasi kebijakan di konsol IAM (konsol)

Anda dapat melihat temuan yang dihasilkan oleh validasi kebijakan IAM Access Analyzer saat membuat atau mengedit kebijakan terkelola di IAM konsol. Anda juga dapat melihat temuan ini untuk pengguna inline atau kebijakan peran. IAMAccess Analyzer tidak menghasilkan temuan ini untuk kebijakan grup inline.

Untuk melihat temuan yang dihasilkan oleh pemeriksaan kebijakan untuk IAM JSON kebijakan

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Mulai membuat atau mengedit kebijakan dengan salah satu metode berikut:
   a. Untuk membuat kebijakan terkelola baru, buka halaman Kebijakan buat kebijakan baru. Untuk informasi selengkapnya, lihat Membuat kebijakan menggunakan JSON editor.
   b. Untuk melihat pemeriksaan kebijakan untuk kebijakan terkelola pelanggan yang sudah ada, buka halaman Kebijakan, pilih nama kebijakan, lalu pilih Edit. Untuk informasi selengkapnya, lihat Menyunting kebijakan terkelola pelanggan (konsol).
   c. Untuk melihat pemeriksaan kebijakan untuk kebijakan sebaris pada pengguna atau peran, buka halaman Pengguna atau Peran, pilih nama pengguna atau peran, pilih nama kebijakan di tab Izin, lalu pilih Edit. Untuk informasi selengkapnya, lihat Mengedit kebijakan inline (konsol).
3. Di editor kebijakan, pilih JSONtab.
4. Di panel validasi kebijakan di bawah kebijakan, pilih salah satu atau beberapa tab berikut. Nama tab juga menunjukkan jumlah setiap jenis temuan untuk kebijakan Anda.
   - Keamanan — Lihat peringatan jika kebijakan Anda mengizinkan akses yang AWS mempertimbangkan risiko keamanan karena aksesnya terlalu permisif.
   - Kesalahan — Lihat kesalahan jika kebijakan Anda menyertakan baris yang mencegah kebijakan berfungsi.
   - Peringatan — Lihat peringatan jika kebijakan Anda tidak sesuai dengan praktik terbaik, tetapi masalahnya bukan risiko keamanan.
   - Saran — Lihat saran jika AWS merekomendasikan perbaikan yang tidak memengaruhi izin kebijakan.
5. Tinjau detail temuan yang disediakan oleh pemeriksaan kebijakan IAM Access Analyzer. Setiap temuan menunjukkan lokasi masalah yang dilaporkan. Untuk mempelajari lebih lanjut tentang penyebab masalah dan cara mengatasinya, pilih tautan Pelajari selengkapnya di sebelah temuan. Anda juga dapat mencari pemeriksaan kebijakan yang terkait dengan setiap temuan di halaman referensi Pemeriksaan kebijakan Penganalisis Akses.
6. Tidak wajib. Jika Anda mengedit kebijakan yang ada, Anda dapat menjalankan pemeriksaan kebijakan khusus untuk menentukan apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan versi yang ada. Di panel validasi kebijakan di bawah kebijakan, pilih

tab Periksa akses baru, lalu pilih Periksa kebijakan. Jika izin yang dimodifikasi memberikan akses baru, pernyataan akan disorot di panel validasi kebijakan. Jika Anda tidak bermaksud memberikan akses baru, perbarui pernyataan kebijakan dan pilih Periksa kebijakan hingga tidak ada akses baru yang terdeteksi. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan pemeriksaan kebijakan khusus IAM Access Analyzer.

> **Note**
>
> Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

7. Perbarui kebijakan Anda untuk menyelesaikan temuan.

> **Important**
>
> Uji kebijakan baru atau diedit secara menyeluruh sebelum mengimplementasikannya dalam alur kerja produksi Anda.

8. Setelah selesai, pilih Selanjutnya. Validator Kebijakan melaporkan kesalahan sintaks yang tidak dilaporkan oleh IAM Access Analyzer.

> **Note**
>
> Anda dapat beralih antara Visual dan JSONtab kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di tab Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

9. Untuk kebijakan baru, pada halaman Tinjau dan buat, masukkan nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda. Kemudian pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Untuk kebijakan yang ada, pada halaman Tinjau dan simpan, tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda. Pilih Setel versi baru ini sebagai default. centang kotak untuk menyimpan versi yang diperbarui sebagai versi default kebijakan. Kemudian pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Memvalidasi kebijakan menggunakan IAM Access Analyzer (AWS CLI atau) AWS API

Anda dapat melihat temuan yang dihasilkan oleh validasi kebijakan IAM Access Analyzer dari AWS Command Line Interface ()AWS CLI.

Untuk melihat temuan yang dihasilkan oleh validasi kebijakan IAM Access Analyzer (AWS CLI atau) AWS API

Gunakan salah satu langkah berikut:

- AWS CLI: aws accessanalyzer validate-policy
- AWS API: ValidatePolicy

Referensi pemeriksaan kebijakan Access Analyzer

Anda dapat memvalidasi kebijakan Anda menggunakan validasi AWS Identity and Access Management Access Analyzer kebijakan. Anda dapat membuat atau mengedit kebijakan menggunakan AWS CLI, AWS API, atau editor JSON kebijakan di IAM konsol. IAMAccess Analyzer memvalidasi kebijakan Anda terhadap tata bahasa IAM kebijakan dan AWS praktik terbaik. Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan Anda. Temuan ini memberikan rekomendasi yang dapat ditindaklanjuti yang membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik keamanan. Daftar pemeriksaan kebijakan dasar yang disediakan oleh IAM Access Analyzer dibagikan di bawah ini. Tidak ada biaya tambahan yang terkait dengan menjalankan pemeriksaan validasi kebijakan. Untuk mempelajari lebih lanjut tentang memvalidasi kebijakan menggunakan validasi kebijakan, lihat. Validasi kebijakan dengan IAM Access Analyzer

Kesalahan - ARN akun tidak diizinkan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

Menyelesaikan kesalahan

Hapus ID akun dari sumber dayaARN. Sumber daya ARNs untuk beberapa AWS layanan tidak mendukung menentukan ID akun.

Misalnya, Amazon S3 tidak mendukung ID akun sebagai namespace di bucket. ARNs Nama bucket Amazon S3 unik secara global, dan namespace dibagikan oleh semua akun. AWS Untuk melihat semua jenis sumber daya yang tersedia di Amazon S3, lihat Jenis sumber daya yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan.

Istilah terkait

- Sumber daya kebijakan
- Pengidentifikasi Akun
- Sumber Daya ARNs
- AWS sumber daya layanan dengan ARN format

Kesalahan - ARN Wilayah tidak diizinkan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

Menyelesaikan kesalahan

Hapus Wilayah dari sumber dayaARN. Sumber daya ARNs untuk beberapa AWS layanan tidak mendukung menentukan Wilayah.

Misalnya, IAM adalah layanan global. Bagian Wilayah dari sumber IAM daya selalu ARN dikosongkan. IAMsumber daya bersifat global, seperti AWS akun saat ini. Misalnya, setelah masuk sebagai IAM pengguna, Anda dapat mengakses AWS layanan di wilayah geografis mana pun.

- Sumber daya kebijakan
- Sumber Daya ARNs
- AWS sumber daya layanan dengan ARN format

Kesalahan - Ketidakcocokan tipe data

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

Menyelesaikan kesalahan

Perbarui teks untuk menggunakan tipe data yang didukung.

Misalnya, kunci kondisi `Version` global memerlukan tipe `String` data. Jika Anda memberikan tanggal atau bilangan bulat, tipe data tidak akan cocok.

Istilah terkait

- Kunci kondisi global
- IAMJSONelemen kebijakan: Operator kondisi

Kesalahan - Kunci duplikat dengan kasus yang berbeda

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

Menyelesaikan kesalahan

Tinjau kunci kondisi serupa dalam blok kondisi yang sama dan gunakan kapitalisasi yang sama untuk semua instance.

Blok kondisi adalah teks dalam `Condition` elemen pernyataan kebijakan. Nama kunci kondisi tidak peka dengan huruf besar-kecil. Sensitivitas kasus nilai kunci kondisi tergantung pada operator kondisi yang Anda gunakan. Untuk informasi selengkapnya tentang sensitivitas huruf besar/kecil pada tombol kondisi, lihat. IAMJSONelemen kebijakan: Condition

Istilah terkait

- Ketentuan
- Blok kondisi
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan - Tindakan tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

Menyelesaikan kesalahan

Tindakan yang Anda tentukan tidak valid. Ini bisa terjadi jika Anda salah mengetik awalan layanan atau nama tindakan. Untuk beberapa masalah umum, pemeriksaan kebijakan mengembalikan tindakan yang disarankan.

Istilah terkait

- Tindakan kebijakan
- AWS tindakan layanan

AWS kebijakan terkelola dengan kesalahan ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola berikut mencakup tindakan tidak valid dalam pernyataan kebijakan mereka. Tindakan yang tidak valid tidak memengaruhi izin yang diberikan oleh kebijakan. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola, AWS sarankan agar Anda menghapus tindakan yang tidak valid dari kebijakan Anda.

- Sebuah mazonEMRFull AccessPolicy _v2
- CloudWatchSyntheticsFullAccess

Kesalahan - Akun tidak valid ARN

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-
digit account ID."
```

Menyelesaikan kesalahan

Perbarui ID akun di sumber dayaARN. Akun IDs adalah bilangan bulat 12 digit. Untuk mempelajari cara melihat ID akun, lihat Menemukan ID AWS akun.

Istilah terkait

- Sumber daya kebijakan

- Pengidentifikasi Akun
- Sumber Daya ARNs
- AWS sumber daya layanan dengan ARN format

Kesalahan - Awalan tidak valid ARN

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

Menyelesaikan kesalahan

AWS sumber daya ARNs harus menyertakan `arn:` awalan yang diperlukan.

Istilah terkait

- Sumber daya kebijakan
- Sumber Daya ARNs
- AWS sumber daya layanan dengan ARN format

Kesalahan - Wilayah Tidak Valid ARN

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

Menyelesaikan kesalahan

Jenis sumber daya tidak didukung di Wilayah yang ditentukan. Untuk tabel AWS layanan yang didukung di setiap Wilayah, lihat tabel Wilayah.

Istilah terkait

- Sumber daya kebijakan
- Sumber Daya ARNs
- Nama dan kode wilayah

Kesalahan - Sumber daya tidak valid ARN

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

Menyelesaikan kesalahan

Sumber daya ARN harus sesuai dengan spesifikasi untuk jenis sumber daya yang diketahui. Untuk melihat ARN format yang diharapkan untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat jenis dan ARN format sumber dayanya.

Istilah terkait

- Sumber daya kebijakan
- Sumber Daya ARNs
- AWS sumber daya layanan dengan ARN format

Kesalahan - Kasus layanan tidak valid ARN

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN service case: Update the service name ${service} in the resource ARN to use
 all lowercase letters.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Update the service name ${service} in the resource ARN to use all
 lowercase letters."
```

Menyelesaikan kesalahan

Layanan dalam sumber daya ARN harus sesuai dengan spesifikasi (termasuk kapitalisasi) untuk awalan layanan. Untuk melihat awalan untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan dan temukan awalannya di kalimat pertama.

Istilah terkait

- Sumber daya kebijakan
- Sumber Daya ARNs
- AWS sumber daya layanan dengan ARN format

Kesalahan - Tipe data kondisi tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

Menyelesaikan kesalahan

Nilai dalam pasangan kunci-nilai kondisi harus sesuai dengan tipe data dari kunci kondisi dan operator kondisi. Untuk melihat tipe data kunci kondisi untuk layanan, lihat Tindakan, sumber daya,

dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat tombol kondisi untuk layanan tersebut.

Misalnya, kunci kondisi `CurrentTime`global mendukung operator `Date` kondisi. Jika Anda memberikan string atau bilangan bulat untuk nilai di blok kondisi, tipe data tidak akan cocok.

Istilah terkait

- Ketentuan
- Blok kondisi
- IAMJSONelemen kebijakan: Operator kondisi
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan - Format kunci kondisi tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

Menyelesaikan kesalahan

Kunci dalam kondisi pasangan kunci-nilai harus sesuai dengan spesifikasi untuk layanan. Untuk melihat kunci kondisi untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat tombol kondisi untuk layanan tersebut.

Istilah terkait

- Ketentuan
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan - Kondisi tidak valid beberapa Boolean

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

Menyelesaikan kesalahan

Kunci dalam pasangan kunci-nilai kondisi mengharapkan nilai Boolean tunggal. Saat Anda memberikan beberapa nilai Boolean, kecocokan kondisi mungkin tidak mengembalikan hasil yang Anda harapkan.

Untuk melihat kunci kondisi untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat tombol kondisi untuk layanan tersebut.

- Ketentuan
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan - Operator kondisi tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

Menyelesaikan kesalahan

Perbarui kondisi untuk menggunakan operator kondisi yang didukung.

Istilah terkait

- IAMJSONelemen kebijakan: Operator kondisi
- Elemen kondisi
- Ikhtisar JSON kebijakan

Kesalahan - Efek tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

Menyelesaikan kesalahan

Perbarui `Effect` elemen untuk menggunakan efek yang valid. Nilai yang valid untuk `Effect` adalah **`Allow`** dan **`Deny`**.

Istilah terkait

- Elemen efek
- Ikhtisar JSON kebijakan

Kesalahan - Kunci kondisi global tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

Menyelesaikan kesalahan

Perbarui kunci kondisi pada pasangan kunci-nilai kondisi untuk menggunakan kunci kondisi global yang didukung.

Kunci kondisi global adalah kunci kondisi dengan `aws:` awalan. AWS layanan dapat mendukung kunci kondisi global atau menyediakan kunci khusus layanan yang menyertakan awalan layanan mereka. Misalnya, tombol IAM kondisi menyertakan `iam:` awalan. Untuk informasi selengkapnya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan yang kuncinya ingin Anda lihat.

Istilah terkait

- Kunci kondisi global

Kesalahan - Partisi tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

Menyelesaikan kesalahan

Perbarui sumber daya ARN untuk menyertakan partisi yang didukung. Jika Anda menyertakan partisi yang didukung, maka layanan atau sumber daya mungkin tidak mendukung partisi yang Anda sertakan.

Partisi adalah sekelompok AWS Wilayah. Setiap AWS akun dicakup ke satu partisi. Di Wilayah Klasik, gunakan `aws` partisi. Di Wilayah China, gunakan`aws-cn`.

Istilah terkait

- Nama Sumber Daya Amazon (ARNs) - Partisi

Kesalahan - Elemen kebijakan tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid policy element: The policy element {{element}} is not valid.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy element {{element}} is not valid."
```

Menyelesaikan kesalahan

Perbarui kebijakan agar hanya menyertakan elemen JSON kebijakan yang didukung.

Istilah terkait

- JSONelemen kebijakan

Kesalahan - Format utama tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid principal format: The Principal element contents are not valid. Specify a key-
value pair in the Principal element.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value
 pair in the Principal element."
```

Menyelesaikan kesalahan

Perbarui prinsipal untuk menggunakan format pasangan nilai kunci yang didukung.

Anda dapat menentukan prinsipal dalam kebijakan berbasis sumber daya, tetapi bukan kebijakan berbasis identitas.

Misalnya, untuk menentukan akses bagi semua orang di AWS akun, gunakan prinsip berikut dalam kebijakan Anda:

```
"Principal": { "AWS": "123456789012" }
```

Istilah terkait

- JSONelemen kebijakan: Principal
- Kebijakan berbasis identitas dan kebijakan berbasis sumber daya

Kesalahan - Kunci utama tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Menyelesaikan kesalahan

Perbarui kunci dalam pasangan nilai kunci utama untuk menggunakan kunci utama yang didukung. Berikut ini adalah kunci utama yang didukung:

- AWS
- CanonicalUser
- Federasi
- Layanan

Istilah terkait

- Elemen utama

Kesalahan - Wilayah Tidak Valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 suported Region.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 suported Region."
```

Menyelesaikan kesalahan

Perbarui nilai pasangan kunci-nilai kondisi untuk menyertakan Wilayah yang didukung. Untuk tabel AWS layanan yang didukung di setiap Wilayah, lihat tabel Wilayah.

Istilah terkait

- Sumber daya kebijakan
- Sumber Daya ARNs
- Nama dan kode wilayah

Kesalahan - Layanan tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

Menyelesaikan kesalahan

Awalan layanan dalam kunci tindakan atau kondisi harus sesuai dengan spesifikasi (termasuk kapitalisasi) untuk awalan layanan. Untuk melihat awalan untuk layanan, lihat Tindakan, sumber

daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan dan temukan awalannya di kalimat pertama.

Istilah terkait

- Layanan yang dikenal dan tindakan, sumber daya, dan kunci kondisinya

Kesalahan - Kunci kondisi layanan tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

Menyelesaikan kesalahan

Perbarui kunci dalam pasangan kunci-nilai kondisi untuk menggunakan kunci kondisi yang diketahui untuk layanan. Nama kunci kondisi global dimulai dengan `aws` awalan. AWS layanan dapat menyediakan kunci khusus layanan yang menyertakan awalan layanan mereka. Untuk melihat awalan untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan.

Istilah terkait

- Kunci kondisi global
- Layanan yang dikenal dan tindakan, sumber daya, dan kunci kondisinya

Kesalahan - Layanan tidak valid dalam tindakan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

Menyelesaikan kesalahan

Awalan layanan dalam tindakan harus sesuai dengan spesifikasi (termasuk kapitalisasi) untuk awalan layanan. Untuk melihat awalan untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan dan temukan awalannya di kalimat pertama.

Istilah terkait

- Elemen aksi
- Layanan yang dikenal dan tindakan mereka

Kesalahan - Variabel tidak valid untuk operator

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

Menyelesaikan kesalahan

Anda dapat menggunakan variabel kebijakan di elemen `Resource` dan dalam perbandingan string dalam elemen `Condition`. Kondisi mendukung variabel ketika Anda menggunakan operator string atau ARN operator. Operator string termasuk`StringEquals`,`StringLike`, dan`StringNotLike`. ARNoperator termasuk `ArnEquals` dan`ArnLike`. Anda tidak dapat menggunakan variabel kebijakan dengan operator lain, seperti Numerik, Tanggal, Boolean, Biner, Alamat IP, atau operator Null.

Istilah terkait

- Menggunakan variabel kebijakan dalam elemen Kondisi

- Elemen kondisi

Kesalahan - Versi tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid version: The version ${version} is not valid. Use one of the following
 versions: ${versions}
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The version ${version} is not valid. Use one of the following
 versions: ${versions}"
```

Menyelesaikan kesalahan

Elemen `Version` kebijakan menentukan aturan sintaks bahasa yang AWS digunakan untuk memproses kebijakan. Untuk menggunakan semua fitur kebijakan yang tersedia, sertakan `Version` elemen terbaru sebelum `Statement` elemen di semua kebijakan Anda.

```
"Version": "2012-10-17"
```

Istilah terkait

- Elemen versi

Kesalahan - Kesalahan sintaks Json

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

Menyelesaikan kesalahan

Kebijakan Anda menyertakan kesalahan sintaks. Periksa JSON sintaks Anda.

Istilah terkait

- JSONvalidator
- IAMJSONreferensi elemen kebijakan
- Ikhtisar JSON kebijakan

Kesalahan - Kesalahan sintaks Json

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Json syntax error: Fix the JSON syntax error.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Fix the JSON syntax error."
```

Menyelesaikan kesalahan

Kebijakan Anda menyertakan kesalahan sintaks. Periksa JSON sintaks Anda.

Istilah terkait

- JSONvalidator
- IAMJSONreferensi elemen kebijakan
- Ikhtisar JSON kebijakan

Kesalahan - Tindakan hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing action: Add an Action or NotAction element to the policy statement.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

Menyelesaikan kesalahan

AWS JSONkebijakan harus mencakup `NotAction` elemen `Action` atau.

Istilah terkait

- Elemen aksi
- NotAction elemen
- Ikhtisar JSON kebijakan

Kesalahan - ARN Bidang hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

Menyelesaikan kesalahan

Semua bidang dalam sumber daya ARN harus sesuai dengan spesifikasi untuk jenis sumber daya yang diketahui. Untuk melihat ARN format yang diharapkan untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat jenis dan ARN format sumber dayanya.

Istilah terkait

- Sumber daya kebijakan
- Sumber Daya ARNs

- AWS sumber daya layanan dengan ARN format

Kesalahan - ARN Wilayah Hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

Menyelesaikan kesalahan

Sumber daya ARNs untuk sebagian besar AWS layanan mengharuskan Anda menentukan Wilayah. Untuk tabel AWS layanan yang didukung di setiap Wilayah, lihat tabel Wilayah.

Istilah terkait

- Sumber daya kebijakan
- Sumber Daya ARNs
- Nama dan kode wilayah

Kesalahan - Efek hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

Menyelesaikan kesalahan

AWS JSONkebijakan harus menyertakan `Effect` elemen dengan nilai **`Allow`** dan**`Deny`**.

Istilah terkait

- Elemen efek
- Ikhtisar JSON kebijakan

Kesalahan - Kepala sekolah yang hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing principal: Add a Principal element to the policy statement.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a Principal element to the policy statement."
```

Menyelesaikan kesalahan

Kebijakan berbasis sumber daya harus menyertakan elemen. `Principal`

Misalnya, untuk menentukan akses bagi semua orang di AWS akun, gunakan prinsip berikut dalam kebijakan Anda:

```
"Principal": { "AWS": "123456789012" }
```

Istilah terkait

- Elemen utama
- Kebijakan berbasis identitas dan kebijakan berbasis sumber daya

Kesalahan - Kualifikasi tidak ada

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing qualifier: The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

Menyelesaikan kesalahan

Dalam `Condition` elemen, Anda membangun ekspresi di mana Anda menggunakan operator kondisi seperti sama atau kurang dari untuk membandingkan kondisi dalam kebijakan terhadap kunci dan nilai dalam konteks permintaan. Untuk permintaan yang menyertakan beberapa nilai untuk satu kunci kondisi, Anda harus menyertakan kondisi dalam tanda kurung seperti array (“Key2": [" Value2A”, “Value2B"]). Anda juga harus menggunakan `ForAllValues` atau `ForAnyValue` mengatur operator dengan operator `StringLike` kondisi. Pengkualifikasi ini menambahkan fungsi operasi kumpulan ke operator kondisi sehingga Anda dapat menguji beberapa nilai permintaan terhadap beberapa nilai kondisi.

Istilah terkait

- Kunci konteks multivaluasi
- Elemen kondisi

AWS kebijakan terkelola dengan kesalahan ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola berikut menyertakan kualifikasi yang hilang untuk kunci kondisi dalam pernyataan kebijakan mereka. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola pelanggan, AWS sarankan Anda menambahkan `ForAllValues` atau `ForAnyValue` mengkondisikan kualifikasi kunci ke `Condition` elemen Anda.

- AWSGlueConsoleSageMakerNotebookFullAccess

Kesalahan - Sumber daya hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

Menyelesaikan kesalahan

Semua kebijakan kecuali kebijakan kepercayaan peran harus menyertakan `NotResource` elemen `Resource` atau.

Istilah terkait

- Elemen sumber daya
- NotResource elemen
- Kebijakan berbasis identitas dan kebijakan berbasis sumber daya
- Ikhtisar JSON kebijakan

Kesalahan - Pernyataan hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing statement: Add a statement to the policy
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a statement to the policy"
```

Menyelesaikan kesalahan

JSONKebijakan harus menyertakan pernyataan.

Istilah terkait

- JSONelemen kebijakan

Kesalahan - Null dengan jika ada

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

Menyelesaikan kesalahan

Anda dapat menambahkan `IfExists` ke akhir nama operator kondisi apa pun kecuali operator `Null` kondisi. Gunakan operator ketentuan `Null` untuk memeriksa apakah kunci kondisi ada pada saat otorisasi. Gunakan `...ifExists` untuk mengatakan “Jika kunci kebijakan hadir dalam konteks permintaan, proses kunci seperti yang ditentukan dalam kebijakan. Jika kuncinya tidak ada, evaluasi elemen ketentuan sebagai benar."

Istilah terkait

- ... IfExists operator kondisi
- Operator kondisi nol
- Elemen kondisi

Kesalahan - SCP wildcard aksi kesalahan sintaks

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCPs) mendukung menentukan nilai-nilai dalam `Action` atau `NotAction` elemen. Namun, nilai-nilai ini dapat mencakup wildcard (*) hanya di akhir string. Ini berarti Anda dapat menentukan `iam:Get*` tetapi tidak`iam:*role`.

Untuk menentukan beberapa tindakan, AWS sarankan Anda mencantumkannya satu per satu.

Istilah terkait

- SCPAksi dan NotAction elemen
- SCPevaluasi
- AWS Organizations kebijakan kontrol layanan
- IAMJSONelemen kebijakan: Aksi

Kesalahan - kesalahan SCP sintaks memungkinkan kondisi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error allow condition: SCPs do not support the Condition element with effect
 Allow. Update the element Condition or the effect.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support the Condition element with effect Allow. Update
 the element Condition or the effect."
```

Menyelesaikan kesalahan

AWS Organizations service control policies (SCPs) mendukung menentukan nilai dalam `Condition` elemen hanya ketika Anda menggunakan`"Effect": "Deny"`.

Untuk mengizinkan hanya satu tindakan, Anda dapat menolak akses ke semuanya kecuali kondisi yang Anda tentukan menggunakan `...NotEquals` versi operator kondisi. Ini meniadakan perbandingan yang dibuat oleh operator.

Istilah terkait

- SCPElemen kondisi

- SCPevaluasi
- AWS Organizations kebijakan kontrol layanan
- Contoh kebijakan: Menolak akses AWS berdasarkan Wilayah yang diminta
- IAMJSONelemen kebijakan: Operator kondisi
- IAMJSONelemen kebijakan: Kondisi

Kesalahan - kesalahan SCP sintaks memungkinkan NotAction

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error allow NotAction: SCPs do not support NotAction with effect Allow.
 Update the element NotAction or the effect.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support NotAction with effect Allow. Update the element
 NotAction or the effect."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCPs) tidak mendukung penggunaan `NotAction` elemen dengan`"Effect": "Allow"`.

Anda harus menulis ulang logika untuk mengizinkan daftar tindakan, atau untuk menolak setiap tindakan yang tidak terdaftar.

Istilah terkait

- SCPAksi dan NotAction elemen
- SCPevaluasi
- AWS Organizations kebijakan kontrol layanan
- IAMJSONelemen kebijakan: Aksi

Kesalahan - kesalahan SCP sintaks memungkinkan sumber daya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error allow resource: SCPs do not support Resource with effect Allow. Update
 the element Resource or the effect.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support Resource with effect Allow. Update the element
 Resource or the effect."
```

Menyelesaikan kesalahan

AWS Organizations service control policies (SCPs) mendukung menentukan nilai dalam `Resource` elemen hanya ketika Anda menggunakan`"Effect": "Deny"`.

Anda harus menulis ulang logika untuk memungkinkan semua sumber daya, atau untuk menolak setiap sumber daya yang terdaftar.

Istilah terkait

- SCPElemen sumber daya
- SCPevaluasi
- AWS Organizations kebijakan kontrol layanan
- IAMJSONelemen kebijakan: Sumber daya

Kesalahan - kesalahan SCP sintaks NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error NotResource: SCPs do not support the NotResource element. Update the
 policy to use Resource instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support the NotResource element. Update the policy to
 use Resource instead."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCPs) tidak mendukung `NotResource` elemen.

Anda harus menulis ulang logika untuk memungkinkan semua sumber daya, atau untuk menolak setiap sumber daya yang terdaftar.

Istilah terkait

- SCPElemen sumber daya
- SCPevaluasi
- AWS Organizations kebijakan kontrol layanan
- IAMJSONelemen kebijakan: Sumber daya

Kesalahan - prinsip kesalahan SCP sintaks

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCPs) tidak mendukung `Principal` atau `NotPrincipal` elemen.

Anda dapat menentukan Amazon Resource Name (ARN) menggunakan kunci kondisi `aws:PrincipalArn` global dalam `Condition` elemen.

Istilah terkait

- SCPsintaks
- Kunci kondisi global untuk kepala sekolah

Kesalahan - Sids Unik diperlukan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

Menyelesaikan kesalahan

Untuk beberapa jenis kebijakan, pernyataan IDs harus unik. Elemen `Sid` (ID pernyataan) memungkinkan Anda memasukkan pengenal opsional yang Anda berikan untuk pernyataan kebijakan. Anda dapat menetapkan nilai ID pernyataan untuk setiap pernyataan dalam array pernyataan menggunakan `SID` elemen. Dalam layanan yang memungkinkan Anda menentukan elemen ID, seperti SQS danSNS, `Sid` nilainya hanyalah sub-ID dari ID dokumen kebijakan. Misalnya, dalamIAM, `Sid` nilainya harus unik dalam suatu JSON kebijakan.

Istilah terkait

- IAMJSONelemen kebijakan: Sid

Kesalahan — Tindakan yang tidak didukung dalam kebijakan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa tindakan tidak didukung dalam `Action` elemen dalam kebijakan berbasis sumber daya yang dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS Key Management Service tindakan tidak didukung dalam kebijakan bucket Amazon S3. Tentukan tindakan yang didukung oleh jenis sumber daya yang dilampirkan pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- JSONelemen kebijakan: Aksi

Kesalahan - Kombinasi elemen yang tidak didukung

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported element combination: The policy elements ${element1} and ${element2} can
 not be used in the same statement. Remove one of these elements.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy elements ${element1} and ${element2} can not be used in
 the same statement. Remove one of these elements."
```

Menyelesaikan kesalahan

Beberapa kombinasi elemen JSON kebijakan tidak dapat digunakan bersama. Misalnya, Anda tidak dapat menggunakan `Action` dan `NotAction` dalam pernyataan kebijakan yang sama. Pasangan lain yang saling eksklusif termasuk `Principal/NotPrincipal` dan`Resource/NotResource`.

Istilah terkait

- IAMJSONreferensi elemen kebijakan
- Ikhtisar JSON kebijakan

Kesalahan - Kunci kondisi global yang tidak didukung

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

Menyelesaikan kesalahan

AWS tidak mendukung penggunaan kunci kondisi global yang ditentukan. Tergantung pada kasus penggunaan Anda, Anda dapat menggunakan `aws:PrincipalArn` atau kunci kondisi `aws:SourceArn` global. Misalnya`aws:ARN`, gunakan `aws:PrincipalArn` untuk membandingkan Amazon Resource Name (ARN) prinsipal yang membuat permintaan dengan ARN yang Anda tentukan dalam kebijakan. Atau, gunakan kunci kondisi `aws:SourceArn` global untuk membandingkan Amazon Resource Name (ARN) sumber daya yang membuat service-to-service permintaan dengan ARN yang Anda tentukan dalam kebijakan.

Istilah terkait

- AWS kunci konteks kondisi global

Kesalahan — Prinsipal yang tidak didukung

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported principal: The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element."
```

Menyelesaikan kesalahan

`Principal`Elemen menentukan prinsipal yang diizinkan atau ditolak akses ke sumber daya. Anda tidak dapat menggunakan `Principal` elemen dalam kebijakan IAM berbasis identitas. Anda dapat menggunakannya dalam kebijakan kepercayaan untuk IAM peran dan kebijakan berbasis sumber

daya. Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya. Misalnya, Anda dapat menyematkan kebijakan di bucket Amazon S3 atau AWS KMS kunci.

Istilah terkait

- AWS JSONelemen kebijakan: Principal
- Akses sumber daya lintas akun di IAM

Kesalahan — Sumber daya yang tidak didukung ARN dalam kebijakan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa sumber daya ARNs tidak didukung dalam `Resource` elemen kebijakan berbasis sumber daya saat kebijakan dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS KMS ARNs tidak didukung dalam `Resource` elemen untuk kebijakan bucket Amazon S3. Tentukan sumber daya ARN yang didukung oleh jenis sumber daya yang dilampirkan pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- JSONelemen kebijakan: Aksi

Kesalahan — Sid Tidak Didukung

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported Sid: Update the characters in the Sid element to use one of the following character types: [a-z, A-Z, 0-9]
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]"
```

Menyelesaikan kesalahan

Elemen `Sid` mendukung huruf besar, huruf kecil, dan angka.

Istilah terkait

- IAMJSONelemen kebijakan: Sid

Kesalahan - Wildcard tidak didukung pada prinsipnya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

Menyelesaikan kesalahan

Struktur `Principal` elemen mendukung penggunaan pasangan kunci-nilai. Nilai pokok yang ditentukan dalam polis termasuk wildcard (*). Anda tidak dapat menyertakan wildcard dengan kunci utama yang Anda tentukan. Misalnya, saat Anda menentukan pengguna dalam `Principal` elemen, Anda tidak dapat menggunakan wildcard yang berarti “semua pengguna”. Anda harus memberi nama pengguna atau pengguna tertentu. Demikian pula, ketika Anda menentukan sesi peran yang diasumsikan, Anda tidak dapat menggunakan wildcard yang berarti “semua sesi”. Anda harus menyebutkan sesi tertentu. Anda juga tidak dapat menggunakan wildcard untuk mencocokkan bagian dari nama atau nama. ARN

Untuk mengatasi temuan ini, hapus wildcard dan berikan prinsipal yang lebih spesifik.

Istilah terkait

- AWS JSONelemen kebijakan: Principal

Kesalahan - Penjepit tidak ada dalam variabel

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

Menyelesaikan kesalahan

Struktur variabel kebijakan mendukung penggunaan $ awalan diikuti oleh sepasang kurawal kurawal (). { } Di dalam ${ } karakter, sertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan.

Untuk mengatasi temuan ini, tambahkan penjepit yang hilang untuk memastikan set kawat gigi pembukaan dan penutup penuh hadir.

Istilah terkait

- IAMelemen kebijakan: Variabel

Kesalahan - Kutipan hilang dalam variabel

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

Menyelesaikan kesalahan

Ketika Anda menambahkan variabel ke kebijakan Anda, Anda dapat menentukan nilai default untuk variabel. Jika variabel tidak ada, AWS gunakan teks default yang Anda berikan.

Untuk menambahkan nilai default ke variabel, sertai nilai default dengan tanda kutip tunggal (`' '`), dan pisahkan teks variabel serta nilai default dengan koma dan spasi (`, `).

Misalnya, jika prinsipal diberi tag`team=yellow`, mereka dapat mengakses bucket `amzn-s3-demo-bucket` Amazon S3 dengan nama tersebut. `amzn-s3-demo-bucket-yellow` Kebijakan dengan sumber daya ini mungkin memungkinkan anggota tim untuk mengakses sumber daya mereka sendiri, tetapi bukan milik tim lain. Untuk pengguna tanpa tag tim, Anda dapat menetapkan nilai default`company-wide`. Pengguna ini hanya dapat mengakses `amzn-s3-demo-bucket-company-wide` bucket di mana mereka dapat melihat informasi yang luas, seperti instruksi untuk bergabung dengan tim.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Istilah terkait

- IAMelemen kebijakan: Variabel

Kesalahan - Ruang yang tidak didukung dalam variabel

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

Menyelesaikan kesalahan

Struktur variabel kebijakan mendukung penggunaan $ awalan diikuti oleh sepasang kurawal kurawal (). { } Di dalam ${ } karakter, sertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan. Meskipun Anda dapat menyertakan spasi ketika Anda menentukan variabel default, Anda tidak dapat menyertakan spasi dalam nama variabel.

Istilah terkait

- IAMelemen kebijakan: Variabel

Kesalahan - Variabel kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

Menyelesaikan kesalahan

Struktur variabel kebijakan mendukung penggunaan $ awalan diikuti oleh sepasang kurawal kurawal (). { } Di dalam ${ } karakter, sertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan.

Istilah terkait

- IAMelemen kebijakan: Variabel

Kesalahan - Variabel tidak didukung dalam elemen

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

Menyelesaikan kesalahan

Anda dapat menggunakan variabel kebijakan di elemen `Resource` dan dalam perbandingan string dalam elemen `Condition`.

Istilah terkait

- IAMelemen kebijakan: Variabel

Kesalahan - Variabel tidak didukung dalam versi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

Menyelesaikan kesalahan

Untuk menggunakan variabel kebijakan, Anda harus menyertakan `Version` elemen dan menyetelnya ke versi yang mendukung variabel kebijakan. Variabel diperkenalkan dalam versi `2012-10-17`. Versi sebelumnya dari bahasa kebijakan tidak mendukung variabel kebijakan. Jika Anda tidak menyetel `Version` ke `2012-10-17` atau yang lebih baru, variabel seperti `${aws:username}` diperlakukan sebagai string literal dalam kebijakan.

Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan, dibuat saat Anda mengubah kebijakan yang dikelola pelangganIAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola.

Istilah terkait

- IAMelemen kebijakan: Variabel
- IAMJSONelemen kebijakan: Versi

Kesalahan - Alamat IP pribadi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

Menyelesaikan kesalahan

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik. Anda menerima kesalahan ini ketika kebijakan Anda hanya mengizinkan alamat IP pribadi. Dalam hal ini, kondisinya tidak akan pernah cocok.

- aws: kunci kondisi SourceIp global
- IAMJSONelemen kebijakan: Kondisi

Kesalahan - Pribadi NotIpAddress

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

Menyelesaikan kesalahan

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik. Anda menerima kesalahan ini ketika Anda menggunakan operator `NotIpAddress` kondisi dan hanya mencantumkan alamat IP pribadi. Dalam hal ini, kondisinya akan selalu cocok dan tidak akan efektif.

- aws: kunci kondisi SourceIp global
- IAMJSONelemen kebijakan: Kondisi

Kesalahan — Ukuran kebijakan melebihi SCP kuota

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCPs) mendukung menentukan nilai-nilai dalam `Action` atau `NotAction` elemen. Namun, nilai-nilai ini dapat mencakup wildcard (*) hanya di akhir string. Ini berarti Anda dapat menentukan `iam:Get*` tetapi tidak`iam:*role`.

Untuk menentukan beberapa tindakan, AWS sarankan Anda mencantumkannya satu per satu.

Istilah terkait

- Kuota untuk Organizations AWS

- AWS Organizations kebijakan kontrol layanan

Kesalahan - Format utama layanan tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

Menyelesaikan kesalahan

Nilai dalam pasangan kunci-nilai kondisi harus cocok dengan format utama layanan yang ditentukan.

Pengguna utama layanan adalah pengidentifikasi yang digunakan untuk memberikan izin layanan. Anda dapat menentukan prinsip layanan dalam `Principal` elemen atau sebagai nilai untuk beberapa kunci kondisi global dan kunci khusus layanan. Prinsipal layanan ditentukan oleh masing-masing layanan.

Pengidentifikasi untuk kepala layanan mencakup nama layanan, dan biasanya dalam format berikut dalam semua huruf kecil:

`service-name``.amazonaws.com`

Beberapa kunci khusus layanan mungkin menggunakan format yang berbeda untuk prinsipal layanan. Misalnya, kunci `kms:ViaService` kondisi memerlukan format berikut untuk prinsip layanan dalam semua huruf kecil:

`service-name.AWS_region``.amazonaws.com`

Istilah terkait

- Prinsipal layanan
- AWS kunci kondisi global
- `kms:ViaService`kunci kondisi

Kesalahan - Kunci tag tidak ada dalam kondisi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

Menyelesaikan kesalahan

Untuk mengontrol akses berdasarkan tanda, Anda memberikan informasi tanda di elemen ketentuan dari kebijakan.

Misalnya, untuk mengontrol akses ke AWS sumber daya, Anda menyertakan kunci `aws:ResourceTag` kondisi. Kunci ini membutuhkan format`aws:ResourceTag/`*`tag-key`*. Untuk menentukan kunci tag `owner` dan nilai tag `JaneDoe` dalam suatu kondisi, gunakan format berikut.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
}
```

Istilah terkait

- Mengontrol akses menggunakan tag
- Ketentuan
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan - Format vpc tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

Menyelesaikan kesalahan

Kunci `aws:SourceVpc` kondisi harus menggunakan awalan `vpc-` diikuti oleh 8 atau 17 karakter alfanumerik, misalnya, atau. `vpc-11223344556677889` `vpc-12345678`

Istilah terkait

- AWS kunci kondisi global: aws: SourceVpc

Kesalahan - Format vpce tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

Menyelesaikan kesalahan

Kunci `aws:SourceVpce` kondisi harus menggunakan awalan `vpce-` diikuti oleh 8 atau 17 karakter alfanumerik, misalnya, atau. `vpce-11223344556677889` `vpce-12345678`

Istilah terkait

- AWS kunci kondisi global: aws: SourceVpce

Kesalahan - Prinsipal federasi tidak didukung

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

Menyelesaikan kesalahan

`Principal`Elemen ini menggunakan prinsip-prinsip federasi untuk kebijakan kepercayaan yang melekat pada IAM peran untuk menyediakan akses melalui federasi identitas. Kebijakan identitas dan kebijakan berbasis sumber daya lainnya tidak mendukung penyedia identitas federasi di elemen tersebut. `Principal` Misalnya, Anda tidak dapat menggunakan SAML prinsipal dalam kebijakan bucket Amazon S3. Ubah `Principal` elemen ke tipe utama yang didukung.

Istilah terkait

- Menciptakan peran untuk federasi identitas
- JSONelemen kebijakan: Principal

Kesalahan - Tindakan yang tidak didukung untuk kunci kondisi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
 actions. We recommend that you move these actions to a different statement without
 this condition key.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
```

```
recommend that you move these actions to a different statement without this condition
key."
```

Menyelesaikan kesalahan

Pastikan bahwa kunci kondisi dalam `Condition` elemen pernyataan kebijakan berlaku untuk setiap tindakan dalam `Action` elemen. Untuk memastikan bahwa tindakan yang Anda tentukan diizinkan atau ditolak secara efektif oleh kebijakan Anda, Anda harus memindahkan tindakan yang tidak didukung ke pernyataan lain tanpa kunci kondisi.

> **Note**
>
> Jika `Action` elemen memiliki tindakan dengan wildcard, IAM Access Analyzer tidak mengevaluasi tindakan tersebut untuk kesalahan ini.

Istilah terkait

- JSONelemen kebijakan: Aksi

Kesalahan — Tindakan yang tidak didukung dalam kebijakan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa tindakan tidak didukung dalam `Action` elemen dalam kebijakan berbasis sumber daya yang dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS Key Management Service tindakan tidak didukung dalam kebijakan bucket Amazon S3. Tentukan tindakan yang didukung oleh jenis sumber daya yang dilampirkan pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- JSONelemen kebijakan: Aksi

Kesalahan — Sumber daya yang tidak didukung ARN dalam kebijakan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa sumber daya ARNs tidak didukung dalam `Resource` elemen kebijakan berbasis sumber daya saat kebijakan dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS KMS ARNs tidak didukung dalam `Resource` elemen untuk kebijakan bucket Amazon S3. Tentukan sumber daya ARN yang didukung oleh jenis sumber daya yang dilampirkan pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- JSONelemen kebijakan: Aksi

Kesalahan - Kunci kondisi yang tidak didukung untuk prinsipal layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

Menyelesaikan kesalahan

Anda dapat menentukan Layanan AWS dalam `Principal` elemen kebijakan berbasis sumber daya menggunakan prinsip layanan, yang merupakan pengenal untuk layanan. Anda tidak dapat menggunakan beberapa kunci kondisi dengan prinsip layanan tertentu. Misalnya, Anda tidak dapat menggunakan kunci `aws:PrincipalOrgID` kondisi dengan kepala layanan`cloudfront.amazonaws.com`. Anda harus menghapus kunci kondisi yang tidak berlaku untuk prinsip layanan dalam `Principal` elemen.

Istilah terkait

- Prinsipal layanan
- JSONelemen kebijakan: Principal

Kesalahan — Kesalahan sintaks kebijakan kepercayaan peran notprincipal

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

Menyelesaikan kesalahan

Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya yang melekat pada suatu peran. IAM Kebijakan kepercayaan menentukan entitas prinsipal mana (akun, pengguna, peran, dan pengguna gabungan) yang dapat memegang peran tersebut. Kebijakan kepercayaan peran tidak mendukung`NotPrincipal`. Perbarui kebijakan untuk menggunakan `Principal` elemen sebagai gantinya.

Istilah terkait

- JSONelemen kebijakan: Principal
- JSONelemen kebijakan: NotPrincipal

Kesalahan — Kebijakan kepercayaan peran tidak didukung wildcard pada prinsipnya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

Menyelesaikan kesalahan

Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya yang melekat pada suatu peran. IAM Kebijakan kepercayaan menentukan entitas utama mana (akun, pengguna, peran, dan pengguna federasi) yang dapat mengambil peran tersebut. `"Principal:" "*"`tidak didukung dalam `Principal` elemen kebijakan kepercayaan peran. Ganti wildcard dengan nilai pokok yang valid.

Istilah terkait

- JSONelemen kebijakan: Principal

Kesalahan - Sumber kesalahan sintaks kebijakan kepercayaan peran

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

Menyelesaikan kesalahan

Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya yang melekat pada suatu peran. IAM Kebijakan kepercayaan menentukan entitas prinsipal mana (akun, pengguna, peran, dan pengguna gabungan) yang dapat memegang peran tersebut. Kebijakan kepercayaan peran berlaku untuk peran yang melekat padanya. Anda tidak dapat menentukan `NotResource` elemen `Resource` atau dalam kebijakan kepercayaan peran. Hapus `NotResource` elemen `Resource` atau.

- JSONelemen kebijakan: Sumber daya
- JSONelemen kebijakan: NotResource

Kesalahan - Ketik rentang IP ketidakcocokan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

Menyelesaikan kesalahan

Perbarui teks untuk menggunakan tipe data operator kondisi alamat IP, dalam CIDR format.

Istilah terkait

- Operator kondisi alamat IP
- IAMJSONelemen kebijakan: Operator kondisi

Kesalahan - Tindakan tidak ada untuk kunci kondisi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

Menyelesaikan kesalahan

Kunci kondisi dalam `Condition` elemen pernyataan kebijakan tidak dievaluasi kecuali tindakan yang ditentukan ada dalam `Action` elemen. Untuk memastikan bahwa kunci kondisi yang Anda tentukan diizinkan atau ditolak secara efektif oleh kebijakan Anda, tambahkan tindakan ke `Action` elemen.

Istilah terkait

- JSONelemen kebijakan: Aksi

Kesalahan - Sintaks utama federasi tidak valid dalam kebijakan kepercayaan peran

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

Menyelesaikan kesalahan

Nilai pokok menentukan prinsip federasi yang tidak sesuai dengan format yang diharapkan. Perbarui format prinsipal federasi ke nama domain atau SAML ARN metadata yang valid.

Istilah terkait

- Pengguna dan peran federasi

Kesalahan - Tindakan tidak cocok untuk prinsipal

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

Menyelesaikan kesalahan

Tindakan yang ditentukan dalam `Action` elemen pernyataan kebijakan tidak valid dengan prinsipal yang ditentukan dalam elemen. `Principal` Misalnya, Anda tidak dapat menggunakan prinsipal SAML penyedia dengan `sts:AssumeRoleWithWebIdentity` tindakan tersebut. Anda harus menggunakan prinsipal SAML penyedia dengan `sts:AssumeRoleWithSAML` tindakan atau menggunakan prinsip OIDC penyedia dengan `sts:AssumeRoleWithWebIdentity` tindakan tersebut.

Istilah terkait

- AssumeRoleWithSAML

- AssumeRoleWithWebIdentity

Kesalahan - Tindakan tidak ada untuk peran di mana saja kebijakan kepercayaan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

Menyelesaikan kesalahan

Agar IAM Roles Anywhere dapat mengambil peran dan memberikan AWS kredensi sementara, peran tersebut harus mempercayai prinsip layanan IAM Roles Anywhere. Prinsipal layanan IAM Roles Anywhere memerlukan `sts:AssumeRole`,`sts:SetSourceIdentity`, dan `sts:TagSession` izin untuk mengambil peran. Jika ada izin yang hilang, Anda harus menambahkannya ke kebijakan Anda.

Istilah terkait

- Model kepercayaan dalam AWS Identity and Access Management Peran Di Mana Saja

Peringatan Umum - Buat SLR dengan NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat IAM peran yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `iam:CreateServiceLinkedRole` kebijakan dengan `NotResource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- IAMJSONelemen kebijakan: NotResource
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Umum - Buat SLR dengan bintang beraksi dan NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat IAM peran yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Kebijakan dengan wildcard

(*) dalam `Action` dan yang menyertakan `NotResource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- IAMJSONelemen kebijakan: NotResource
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Umum - Buat SLR dengan NotAction dan NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat IAM peran yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `NotAction` elemen dengan `NotResource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menulis ulang kebijakan untuk mengizinkan `iam:CreateServiceLinkedRole` pada daftar terbatas ARNs dalam `Resource` elemen sebagai gantinya. Anda juga dapat `iam:CreateServiceLinkedRole` menambahkan `NotAction` elemen.

- CreateServiceLinkedRole operasi
- IAMJSONelemen kebijakan: NotAction
- IAMJSONelemen kebijakan: Aksi

- IAMJSONelemen kebijakan: NotResource
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Umum - Buat SLR dengan bintang di sumber daya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat IAM peran yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `iam:CreateServiceLinkedRole` dalam kebijakan dengan wildcard (*) dalam `Resource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- IAMJSONelemen kebijakan: Sumber daya

AWS kebijakan terkelola dengan peringatan umum ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk pengguna yang kuat di dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses pengguna yang kuat dan memberikan izin untuk membuat peran terkait layanan untuk layanan apa pun. AWS AWS merekomendasikan agar

Anda melampirkan kebijakan AWS terkelola berikut hanya IAM pada identitas yang Anda anggap pengguna daya.

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy— Kebijakan AWS terkelola ini memberikan izin untuk digunakan oleh peran AWS Organizations terkait layanan. Peran ini memungkinkan Organizations untuk membuat peran terkait layanan tambahan untuk layanan lain di organisasi Anda AWS .

Peringatan Umum - Buat SLR dengan bintang dalam aksi dan sumber daya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat IAM peran yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Kebijakan dengan wildcard (*) di `Resource` elemen `Action` dan dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. Hal ini memungkinkan pembuatan peran terkait layanan saat Anda menentukan`"Action": "*"`,`"Action": "iam:*"`, atau. `"Action": "iam:Create*"` AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- IAMJSONelemen kebijakan: Tindakan

- IAMJSONelemen kebijakan: Sumber daya

AWS kebijakan terkelola dengan peringatan umum ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses administrator dan memberikan izin untuk membuat peran terkait layanan untuk layanan apa pun. AWS AWS merekomendasikan agar Anda melampirkan kebijakan AWS terkelola berikut hanya pada IAM identitas yang Anda anggap administrator.

- AdministratorAccess
- IAMFullAccess

Peringatan Umum - Buat SLR dengan bintang di sumber daya dan NotAction

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat IAM peran yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `NotAction` elemen dalam kebijakan dengan wildcard (*) dalam `Resource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya. Anda juga dapat `iam:CreateServiceLinkedRole` menambahkan `NotAction` elemen.

- CreateServiceLinkedRole operasi
- IAMJSONelemen kebijakan: NotAction
- IAMJSONelemen kebijakan: Tindakan
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Umum - Kunci kondisi global yang tidak digunakan lagi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

Menyelesaikan peringatan umum

Kebijakan ini mencakup kunci kondisi global yang tidak digunakan lagi. Perbarui kunci kondisi pada pasangan kunci-nilai kondisi untuk menggunakan kunci kondisi global yang didukung.

- Kunci kondisi global

Peringatan Umum - Nilai tanggal tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

Menyelesaikan peringatan umum

Waktu Unix Epoch menggambarkan titik waktu yang telah berlalu sejak 1 Januari 1970, dikurangi detik kabisat. Waktu zaman mungkin tidak sesuai dengan waktu yang tepat yang Anda harapkan. AWS merekomendasikan agar Anda menggunakan standar W3C untuk format tanggal dan waktu. Misalnya, Anda dapat menentukan tanggal lengkap, seperti YYYY -MM-DD (1997-07-16), atau Anda juga dapat menambahkan waktu ke tanggal kedua, seperti YYYY -MM-:mm:ss (1997-07-16T 19:20:30 + 01:00). DDThh TZD

- Format Tanggal dan Waktu W3C
- IAMJSONelemen kebijakan: Versi
- aws: kunci kondisi CurrentTime global

Peringatan Umum - Referensi peran tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

Menyelesaikan peringatan umum

AWS merekomendasikan agar Anda menentukan Amazon Resource Name (ARN) untuk IAM peran, bukan ID utamanya. Saat IAM menyimpan kebijakan, itu akan mengubah ARN menjadi ID utama untuk peran yang ada. AWS termasuk tindakan pencegahan keamanan. Jika seseorang menghapus dan membuat ulang peran, itu akan memiliki ID baru, dan kebijakan tidak akan cocok dengan ID peran baru.

- Menentukan prinsipal: peran IAM
- IAM ARNs
- IAMunik IDs

Peringatan Umum - Referensi pengguna tidak valid

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

Menyelesaikan peringatan umum

AWS merekomendasikan agar Anda menentukan Amazon Resource Name (ARN) untuk IAM pengguna, bukan ID utamanya. Saat IAM menyimpan kebijakan, itu akan mengubah ARN menjadi ID utama untuk pengguna yang ada. AWS termasuk tindakan pencegahan keamanan. Jika seseorang menghapus dan membuat ulang pengguna, itu akan memiliki ID baru, dan kebijakan tidak akan cocok dengan ID pengguna baru.

- Menentukan prinsipal: pengguna IAM
- IAM ARNs
- IAMunik IDs

Peringatan Umum - Versi tidak ada

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

Menyelesaikan peringatan umum

AWS merekomendasikan agar Anda menyertakan `Version` parameter opsional dalam kebijakan Anda. Jika Anda tidak menyertakan elemen Versi, nilai default, tetapi fitur yang lebih baru`2012-10-17`, seperti variabel kebijakan, tidak akan berfungsi dengan kebijakan Anda. Misalnya, variabel seperti `${aws:username}` tidak diakui sebagai variabel dan diperlakukan sebagai string literal di dalam kebijakan.

- IAMJSONelemen kebijakan: Versi

Peringatan Umum - Sids Unik direkomendasikan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

Menyelesaikan peringatan umum

AWS merekomendasikan agar Anda menggunakan pernyataan unikIDs. Elemen `Sid` (ID pernyataan) memungkinkan Anda memasukkan pengenal opsional yang Anda berikan untuk pernyataan kebijakan. Anda dapat menetapkan nilai ID pernyataan untuk setiap pernyataan dalam array pernyataan menggunakan `SID` elemen.

Istilah terkait

- IAMJSONelemen kebijakan: Sid

Peringatan Umum - Wildcard tanpa operator seperti

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

Menyelesaikan peringatan umum

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Bila Anda menentukan nilai kondisi yang menggunakan wildcard (*,?) , Anda harus menggunakan `Like` versi operator kondisi. Misalnya, alih-alih operator kondisi `StringEquals` string, gunakan`StringLike`.

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- IAMJSONelemen kebijakan: Operator kondisi
- IAMJSONelemen kebijakan: Kondisi

AWS kebijakan terkelola dengan peringatan umum ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola berikut menyertakan wildcard dalam nilai kondisinya tanpa operator kondisi yang menyertakan `Like` pencocokan pola. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola pelanggan, AWS sarankan agar Anda menggunakan operator kondisi yang mendukung pencocokan pola dengan wildcard (*,?) , seperti`StringLike`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Peringatan Umum — Ukuran kebijakan melebihi kuota kebijakan identitas

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
 maximum for inline and managed policies. We recommend that you use multiple granular
 policies.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

Menyelesaikan peringatan umum

Anda dapat melampirkan hingga 10 kebijakan terkelola ke IAM identitas (pengguna, grup pengguna, atau peran). Namun, ukuran setiap kebijakan terkelola tidak dapat melebihi kuota default 6.144 karakter. IAMtidak menghitung spasi putih saat menghitung ukuran kebijakan terhadap kuota ini. Kuota, juga disebut sebagai batas dalam AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item di AWS akun Anda.

Selain itu, Anda dapat menambahkan kebijakan sebaris sebanyak yang Anda inginkan ke IAM identitas. Namun, ukuran jumlah semua kebijakan inline per identitas tidak dapat melebihi kuota yang ditentukan.

Jika kebijakan Anda lebih besar dari kuota, Anda dapat mengatur kebijakan Anda menjadi beberapa pernyataan dan mengelompokkan pernyataan ke dalam beberapa kebijakan.

Istilah terkait

- IAMdan kuota AWS STS karakter
- Beberapa pernyataan dan beberapa kebijakan
- IAMkebijakan yang dikelola pelanggan
- Ikhtisar JSON kebijakan
- IAMJSONtata bahasa kebijakan

AWS kebijakan terkelola dengan peringatan umum ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola berikut memberikan izin untuk tindakan di banyak AWS layanan dan melebihi ukuran kebijakan maksimum. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola, Anda harus membagi kebijakan menjadi beberapa kebijakan.

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

Peringatan Umum - Ukuran kebijakan melebihi kuota kebijakan sumber daya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

Menyelesaikan peringatan umum

Kebijakan berbasis sumber daya adalah dokumen JSON kebijakan yang Anda lampirkan ke sumber daya, seperti bucket Amazon S3. Kebijakan ini memberikan izin pokok yang ditentukan untuk melakukan tindakan spesifik pada sumber daya tersebut dan menentukan dalam kondisi apa hal ini berlaku. Ukuran kebijakan berbasis sumber daya tidak dapat melebihi kuota yang ditetapkan untuk sumber daya tersebut. Kuota, juga disebut sebagai batas dalam AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item di AWS akun Anda.

Jika kebijakan Anda lebih besar dari kuota, Anda dapat mengatur kebijakan Anda menjadi beberapa pernyataan dan mengelompokkan pernyataan ke dalam beberapa kebijakan.

Istilah terkait

- Kebijakan berbasis sumber daya
- Kebijakan bucket Amazon S3
- Beberapa pernyataan dan beberapa kebijakan
- Ikhtisar JSON kebijakan
- IAMJSONtata bahasa kebijakan

Peringatan Umum - Ketidakcocokan Jenis

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi yang didukung.

Misalnya, kunci kondisi `aws:MultiFactorAuthPresent` global memerlukan operator kondisi dengan tipe `Boolean` data. Jika Anda memberikan tanggal atau bilangan bulat, tipe data tidak akan cocok.

Istilah terkait

- Kunci kondisi global
- IAMJSONelemen kebijakan: Operator kondisi

Peringatan Umum - Ketik ketidakcocokan Boolean

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi Boolean, seperti `true` atau`false`.

Misalnya, kunci kondisi `aws:MultiFactorAuthPresent` global memerlukan operator kondisi dengan tipe `Boolean` data. Jika Anda memberikan tanggal atau bilangan bulat, tipe data tidak akan cocok.

Istilah terkait

- Operator kondisi boolean
- IAMJSONelemen kebijakan: Operator kondisi

Peringatan Umum - Ketik tanggal ketidakcocokan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi tanggal, dalam format waktu tanggal ISO 8601 `YYYY-MM-DD` atau lainnya.

Istilah terkait

- Operator kondisi tanggal
- IAMJSONelemen kebijakan: Operator kondisi

Peringatan Umum - Ketik nomor ketidakcocokan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi numerik.

Istilah terkait

- Operator kondisi numerik
- IAMJSONelemen kebijakan: Operator kondisi

Peringatan Umum - Ketik string ketidakcocokan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi string.

Istilah terkait

- Operator kondisi string
- IAMJSONelemen kebijakan: Operator kondisi

Peringatan Umum - Direkomendasikan repo dan cabang github khusus

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value of
 token.actions.githubusercontent.com:sub with the repository and branch name."
```

Menyelesaikan peringatan umum

Jika Anda menggunakan GitHub sebagai OIDC IDP, praktik terbaik adalah membatasi entitas yang dapat mengambil peran yang terkait dengan IAM iDP. Saat menyertakan `Condition` pernyataan dalam kebijakan kepercayaan peran, Anda dapat membatasi peran tersebut ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan tombol kondisi `token.actions.githubusercontent.com:sub` untuk membatasi akses. Kami menyarankan Anda membatasi kondisi untuk satu set repositori atau cabang tertentu. Jika Anda menggunakan wildcard (`*`) di`token.actions.githubusercontent.com:sub`, maka GitHub Tindakan dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan iDP di akun GitHub IAM Anda. AWS

Istilah terkait

- Mengkonfigurasi peran untuk penyedia GitHub OIDC identitas

Peringatan Umum - Ukuran kebijakan melebihi kuota kebijakan kepercayaan peran

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
 role trust policy length quota increase using Service Quotas and AWS Support Center.
 If the quotas have already been increased, then you can ignore this warning.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

Menyelesaikan peringatan umum

IAMdan AWS STS memiliki kuota yang membatasi ukuran kebijakan kepercayaan peran. Karakter dalam kebijakan kepercayaan peran, tidak termasuk spasi putih, melebihi maksimum karakter. Kami menyarankan Anda untuk meminta peningkatan kuota kebijakan kepercayaan peran menggunakan Service Quotas dan. AWS Support Center Console

Istilah terkait

- IAMdan AWS STS kuota, persyaratan nama, dan batas karakter

Peringatan Keamanan — Izinkan dengan NotPrincipal

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

Menyelesaikan peringatan keamanan

Menggunakan `"Effect": "Allow"` dengan `NotPrincipal` bisa terlalu permisif. Misalnya, ini dapat memberikan izin kepada prinsipal anonim. AWS merekomendasikan agar Anda menentukan prinsip yang memerlukan akses menggunakan elemen. `Principal` Atau, Anda dapat mengizinkan akses luas dan kemudian menambahkan pernyataan lain yang menggunakan `NotPrincipal` elemen dengan`“Effect”: “Deny”`.

- AWS JSONelemen kebijakan: Principal
- AWS JSONelemen kebijakan: NotPrincipal

Peringatan Keamanan - ForAllValues dengan kunci bernilai tunggal

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

Menyelesaikan peringatan keamanan

AWS merekomendasikan agar Anda menggunakan `ForAllValues` satu-satunya dengan kondisi multivaluasi. Operator `ForAllValues` set menguji apakah nilai setiap anggota dari set permintaan adalah subset dari set kunci kondisi. Kondisi tersebut akan memberikan hasil benar jika setiap nilai kunci dalam permintaan tersebut sesuai dengan setidaknya satu nilai dalam kebijakan. Kondisi ini juga akan memberikan hasil benar jika tidak ada kunci dalam permintaan, atau jika nilai kunci menghasilkan kumpulan data nol, seperti string kosong.

Untuk mempelajari apakah suatu kondisi mendukung nilai tunggal atau beberapa nilai, tinjau halaman Tindakan, sumber daya, dan kunci kondisi untuk layanan. Kunci kondisi dengan awalan tipe `ArrayOf` data adalah kunci kondisi multivalued. Misalnya, Amazon SES mendukung kunci dengan nilai tunggal (`String`) dan tipe data `ArrayOfString` multivalued.

- Kunci konteks multivaluasi

Peringatan Keamanan - Lulus peran dengan NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus memberikan IAM peran ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Menggunakan `iam:PassRole` kebijakan dengan `NotResource` elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- Melewati peran ke layanan
- saya: PassedToService
- IAMJSONelemen kebijakan: NotResource
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Keamanan - Lulus peran dengan bintang beraksi dan NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus memberikan IAM peran ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Kebijakan dengan wildcard (*) dalam `Action` dan yang menyertakan `NotResource` elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- Melewati peran ke layanan
- saya: PassedToService
- IAMJSONelemen kebijakan: NotResource
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Keamanan - Lulus peran dengan NotAction dan NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus memberikan IAM peran ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup

pengguna, atau peran). Menggunakan `NotAction` elemen dan daftar beberapa sumber daya dalam `NotResource` elemen dapat memungkinkan prinsipal Anda untuk mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- Melewati peran ke layanan
- saya: PassedToService
- IAMJSONelemen kebijakan: NotAction
- IAMJSONelemen kebijakan: Tindakan
- IAMJSONelemen kebijakan: NotResource
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Keamanan - Lulus peran dengan bintang di sumber daya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus memberikan IAM peran ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Kebijakan yang memungkinkan `iam:PassRole` dan yang menyertakan wildcard (*) dalam `Resource` elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan

diizinkan ARNs dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

Beberapa AWS layanan menyertakan namespace layanan mereka atas nama peran mereka. Pemeriksaan kebijakan ini mempertimbangkan konvensi ini saat menganalisis kebijakan untuk menghasilkan temuan. Misalnya, sumber daya berikut ARN mungkin tidak menghasilkan temuan:

```
arn:aws:iam::*:role/Service*
```

- Melewati peran ke layanan
- saya: PassedToService
- IAMJSONelemen kebijakan: Sumber daya

AWS kebijakan terkelola dengan peringatan keamanan ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Salah satu kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses administrator dan memberikan izin untuk meneruskan IAM peran apa pun ke layanan apa pun. AWS merekomendasikan agar Anda melampirkan kebijakan AWS terkelola berikut hanya IAM pada identitas yang Anda anggap administrator.

- AdministratorAccess-Memperkuat

Kebijakan AWS terkelola berikut mencakup izin `iam:PassRole` dengan wildcard (*) di sumber daya dan berada di jalur penghentian. Untuk setiap kebijakan ini, kami memperbarui panduan izin, seperti merekomendasikan kebijakan AWS terkelola baru yang mendukung kasus penggunaan. Untuk melihat alternatif kebijakan ini, lihat panduan untuk setiap layanan.

- AWSElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess
- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOpsWorksRole

- AWSDataPipelineRole
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDBFullAccesswithDataPipeline
- Amazon EC2ContainerServiceFullAccess

Kebijakan AWS terkelola berikut memberikan izin hanya untuk peran terkait layanan, yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Anda tidak dapat melampirkan kebijakan ini ke IAM identitas Anda.

- AWSServiceRoleForAmazonEKSNodegroup

Peringatan Keamanan - Lulus peran dengan bintang dalam aksi dan sumber daya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus memberikan IAM peran ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Kebijakan dengan wildcard (*) di `Resource` elemen `Action` dan dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- Melewati peran ke layanan
- saya: PassedToService
- IAMJSONelemen kebijakan: Tindakan
- IAMJSONelemen kebijakan: Sumber daya

AWS kebijakan terkelola dengan peringatan keamanan ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses administrator dan memberikan izin untuk meneruskan IAM peran apa pun ke AWS layanan apa pun. AWS merekomendasikan agar Anda melampirkan kebijakan AWS terkelola berikut hanya pada IAM identitas yang Anda anggap administrator.

- AdministratorAccess
- IAMFullAccess

Peringatan Keamanan - Lulus peran dengan bintang di sumber daya dan NotAction

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus memberikan IAM peran ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup

pengguna, atau peran). Menggunakan `NotAction` elemen dalam kebijakan dengan wildcard (*) dalam `Resource` elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan diizinkan ARNs dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- Melewati peran ke layanan
- saya: PassedToService
- IAMJSONelemen kebijakan: NotAction
- IAMJSONelemen kebijakan: Tindakan
- IAMJSONelemen kebijakan: Sumber daya

Peringatan Keamanan - Kunci kondisi berpasangan tidak ada

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

Menyelesaikan peringatan keamanan

Beberapa tombol kondisi lebih aman saat dipasangkan dengan kunci kondisi terkait lainnya. AWS merekomendasikan agar Anda menyertakan kunci kondisi terkait di blok kondisi yang sama dengan kunci kondisi yang ada. Hal ini membuat izin yang diberikan melalui kebijakan lebih aman.

Misalnya, Anda dapat menggunakan tombol `aws:VpcSourceIp` kondisi untuk membandingkan alamat IP dari mana permintaan dibuat dengan alamat IP yang Anda tentukan dalam kebijakan. AWS

merekomendasikan agar Anda menambahkan kunci `aws:SourceVPC` kondisi terkait. Ini memeriksa apakah permintaan berasal dari VPC yang Anda tentukan dalam kebijakan dan alamat IP yang Anda tentukan.

Istilah terkait

- `aws:VpcSourceIp`kunci kondisi global
- `aws:SourceVPC`kunci kondisi global
- Kunci kondisi global
- Elemen kondisi
- Ikhtisar JSON kebijakan

Peringatan Keamanan - Tolak dengan kunci kondisi tag yang tidak didukung untuk layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Menyelesaikan peringatan keamanan

Menggunakan kunci kondisi tag yang tidak didukung dalam `Condition` elemen kebijakan dengan `"Effect": "Deny"` bisa terlalu permisif, karena kondisi diabaikan untuk layanan tersebut. AWS merekomendasikan agar Anda menghapus tindakan layanan yang tidak mendukung kunci kondisi dan membuat pernyataan lain untuk menolak akses ke sumber daya tertentu untuk tindakan tersebut.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dan tidak didukung oleh tindakan layanan, maka kunci tidak termasuk dalam konteks permintaan. Dalam hal ini, kondisi dalam `Deny` pernyataan selalu kembali `false` dan tindakan tidak pernah ditolak. Ini terjadi bahkan jika sumber daya ditandai dengan benar.

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut () ABAC. Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (). RBAC

> **Note**
>
> Beberapa layanan memungkinkan dukungan untuk kunci `aws:ResourceTag` kondisi untuk subset sumber daya dan tindakan mereka. IAMAccess Analyzer mengembalikan temuan untuk tindakan layanan yang tidak didukung. Misalnya, Amazon S3 mendukung `aws:ResourceTag` subset sumber dayanya. Untuk melihat semua jenis sumber daya yang tersedia di Amazon S3 yang mendukung kunci `aws:ResourceTag` kondisi, lihat Jenis sumber daya yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan.

Misalnya, asumsikan bahwa Anda ingin menolak akses untuk menghapus tag menghapus sumber daya tertentu yang ditandai dengan pasangan nilai kunci. `status=Confidential` Juga asumsikan bahwa AWS Lambda memungkinkan Anda untuk menandai dan menghapus tag sumber daya, tetapi tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk menolak tindakan hapus untuk AWS App Mesh dan AWS Backup jika tag ini ada, gunakan tombol `aws:ResourceTag` kondisi. Untuk Lambda, gunakan konvensi penamaan sumber daya yang menyertakan awalan. `"Confidential"` Kemudian sertakan pernyataan terpisah yang mencegah penghapusan sumber daya dengan konvensi penamaan itu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
            }
        },
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
    ]
}
```

> **Warning**
>
> Jangan gunakan... IfExistsversi operator kondisi sebagai solusi untuk temuan ini. Ini berarti "Tolak tindakan jika kunci ada dalam konteks permintaan dan nilainya cocok. Kalau tidak, tolak tindakannya." Pada contoh sebelumnya, termasuk `lambda:DeleteFunction` tindakan dalam `DenyDeleteSupported` pernyataan dengan `StringEqualsIfExists` operator selalu menyangkal tindakan. Untuk tindakan itu, kuncinya tidak ada dalam konteks, dan setiap upaya untuk menghapus jenis sumber daya tersebut ditolak, terlepas dari apakah sumber daya tersebut diberi tag.

Istilah terkait

- Kunci kondisi global
- Membandingkan ABAC dengan RBAC
- IAMJSONelemen kebijakan: Operator kondisi
- Elemen kondisi
- Ikhtisar JSON kebijakan

Peringatan Keamanan - Tolak NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
 because some service actions are not denied by this statement. This is because the
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

Menyelesaikan peringatan keamanan

Menggunakan kunci kondisi tag dalam `Condition` elemen kebijakan dengan elemen `NotAction` dan `"Effect": "Deny"` bisa terlalu permisif. Kondisi ini diabaikan untuk tindakan layanan yang tidak mendukung kunci kondisi. AWS merekomendasikan agar Anda menulis ulang logika untuk menolak daftar tindakan.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dengan`NotAction`, tindakan layanan baru atau yang sudah ada yang tidak mendukung kunci tidak ditolak. AWS merekomendasikan agar Anda secara eksplisit mencantumkan tindakan yang ingin Anda tolak. IAMAccess Analyzer mengembalikan temuan terpisah untuk tindakan terdaftar yang tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk informasi selengkapnya, lihat Peringatan Keamanan - Tolak dengan kunci kondisi tag yang tidak didukung untuk layanan.

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut () ABAC. Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (). RBAC

Istilah terkait

- Kunci kondisi global
- Membandingkan ABAC dengan RBAC
- IAMJSONelemen kebijakan: Operator kondisi
- Elemen kondisi
- Ikhtisar JSON kebijakan

Peringatan Keamanan — Batasi akses ke prinsipal layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

Menyelesaikan peringatan keamanan

Anda dapat menentukan Layanan AWS dalam `Principal` elemen kebijakan berbasis sumber daya menggunakan prinsip layanan, yang merupakan pengenal untuk layanan. Saat memberikan akses ke kepala layanan untuk bertindak atas nama Anda, batasi akses. Anda dapat mencegah kebijakan yang terlalu permisif dengan menggunakan`aws:SourceArn`,`aws:SourceAccount`,`aws:SourceOrgID`, atau kunci `aws:SourceOrgPaths` kondisi untuk membatasi akses ke sumber tertentu, seperti sumber daya tertentu, ID organisasi ARN Akun AWS, atau jalur organisasi. Membatasi akses membantu Anda mencegah masalah keamanan yang disebut masalah wakil yang bingung.

Istilah terkait

- Layanan AWS kepala sekolah
- AWS kunci kondisi global: aws: SourceAccount

- AWS kunci kondisi global: aws: SourceArn
- AWS kunci kondisi global: aws: SourceOrgId
- AWS kunci kondisi global: aws: SourceOrgPaths
- Masalah wakil yang membingungkan

Peringatan Keamanan - Kunci kondisi tidak ada untuk kepala sekolah oidc

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
 federated OIDC principals to ensure that only the intended identity provider assumes
 the role.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

Menyelesaikan peringatan keamanan

Menggunakan prinsipal Open ID Connect tanpa kondisi bisa terlalu permisif. Tambahkan kunci kondisi dengan awalan yang cocok dengan OIDC prinsip federasi Anda untuk memastikan bahwa hanya penyedia identitas yang dituju yang mengambil peran tersebut.

Istilah terkait

- Membuat peran untuk identitas web atau OpenID Connect Federation (konsol)

Peringatan Keamanan - Kunci kondisi repo github tidak ada

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
```

```
Add the token.actions.githubusercontent.com:sub condition key and specify the branch
and repository name in the value.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
 token.actions.githubusercontent.com:sub condition key and specify the branch and
 repository name in the value."
```

Menyelesaikan peringatan keamanan

Jika Anda menggunakan GitHub sebagai OIDC IDP, praktik terbaik adalah membatasi entitas yang dapat mengambil peran yang terkait dengan IAM iDP. Saat menyertakan `Condition` pernyataan dalam kebijakan kepercayaan peran, Anda dapat membatasi peran tersebut ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan tombol kondisi `token.actions.githubusercontent.com:sub` untuk membatasi akses. Kami menyarankan Anda membatasi kondisi untuk satu set repositori atau cabang tertentu. Jika Anda tidak menyertakan kondisi ini, maka GitHub Tindakan dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan GitHub IAM IDP di akun Anda. AWS

Istilah terkait

- Mengkonfigurasi peran untuk penyedia GitHub OIDC identitas

Saran - Tindakan array kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

Menyelesaikan saran

Pernyataan harus mencakup salah satu `Action` atau `NotAction` elemen yang mencakup serangkaian tindakan. Ketika elemen kosong, pernyataan kebijakan tidak memberikan izin. Tentukan tindakan dalam `Action` elemen.

- IAMJSONelemen kebijakan: Tindakan

Saran - Kondisi array kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

Menyelesaikan saran

Struktur `Condition` elemen opsional mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Ketika nilai kondisi kosong, kondisi kembali `true` dan pernyataan kebijakan tidak memberikan izin. Tentukan nilai kondisi.

- IAMJSONelemen kebijakan: Kondisi

Saran - Kondisi array kosong ForAllValues

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
 the request context is empty, we recommend that you use the Null condition operator
 with the value of true instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

Menyelesaikan saran

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Operator `ForAllValues` set menguji apakah nilai setiap anggota dari set permintaan adalah subset dari set kunci kondisi.

Bila Anda menggunakan `ForAllValues` dengan kunci kondisi kosong, kondisi hanya cocok jika tidak ada kunci dalam permintaan. AWS merekomendasikan bahwa jika Anda ingin menguji apakah konteks permintaan kosong, gunakan operator `Null` kondisi sebagai gantinya.

- Kunci konteks multivaluasi
- Operator kondisi nol
- IAMJSONelemen kebijakan: Kondisi

Saran - Kondisi array kosong ForAnyValue

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

Menyelesaikan saran

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Operator `ForAnyValues` set menguji apakah setidaknya satu anggota dari kumpulan nilai permintaan cocok dengan setidaknya satu anggota dari kumpulan nilai kunci kondisi.

Bila Anda menggunakan `ForAnyValues` dengan kunci kondisi kosong, kondisi tidak pernah cocok. Ini berarti bahwa pernyataan tersebut tidak berpengaruh pada kebijakan tersebut. AWS merekomendasikan agar Anda menulis ulang kondisinya.

- Kunci konteks multivaluasi
- IAMJSONelemen kebijakan: Kondisi

Saran - Kondisi array kosong IfExists

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
 request context is empty, we recommend that you use the Null condition operator with
 the value of true instead.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

Menyelesaikan saran

`...IfExists`Sufiks mengedit operator kondisi. Ini berarti bahwa jika kunci kebijakan hadir dalam konteks permintaan, proses kunci sebagaimana ditentukan dalam kebijakan. Jika kunci tidak ada, evaluasi elemen kondisi sebagai benar.

Bila Anda menggunakan `...IfExists` dengan kunci kondisi kosong, kondisi hanya cocok jika tidak ada kunci dalam permintaan. AWS merekomendasikan bahwa jika Anda ingin menguji apakah konteks permintaan kosong, gunakan operator `Null` kondisi sebagai gantinya.

- ... IfExists operator kondisi
- IAMJSONelemen kebijakan: Kondisi

Saran - Prinsipal array kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Menyelesaikan saran

Anda harus menggunakan `NotPrincipal` elemen `Principal` atau dalam kebijakan kepercayaan untuk IAM peran dan kebijakan berbasis sumber daya. Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya.

Bila Anda menyediakan array kosong dalam `Principal` elemen pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS merekomendasikan agar Anda menentukan prinsip yang harus memiliki akses ke sumber daya.

- IAMJSONelemen kebijakan: Principal
- IAMJSONelemen kebijakan: NotPrincipal

Saran - Sumber daya array kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

Menyelesaikan saran

Pernyataan harus menyertakan elemen `Resource` atau `NotResource`.

Bila Anda menyediakan array kosong dalam elemen sumber daya pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS merekomendasikan agar Anda menentukan Amazon Resource Names (ARNs) untuk sumber daya.

- IAMJSONelemen kebijakan: Sumber daya
- IAMJSONelemen kebijakan: NotResource

Saran - Kondisi objek kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

Menyelesaikan saran

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai.

Ketika Anda memberikan objek kosong dalam elemen kondisi pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. Hapus elemen opsional atau tentukan kondisi.

- IAMJSONelemen kebijakan: Kondisi

Saran - Prinsip objek kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Menyelesaikan saran

Anda harus menggunakan `NotPrincipal` elemen `Principal` atau dalam kebijakan kepercayaan untuk IAM peran dan kebijakan berbasis sumber daya. Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya.

Ketika Anda memberikan objek kosong dalam `Principal` elemen pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS merekomendasikan agar Anda menentukan prinsip yang harus memiliki akses ke sumber daya.

- IAMJSONelemen kebijakan: Principal
- IAMJSONelemen kebijakan: NotPrincipal

Saran - Nilai Sid kosong

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

Menyelesaikan saran

Elemen opsional `Sid` (ID pernyataan) memungkinkan Anda memasukkan pengenal yang Anda berikan untuk pernyataan kebijakan. Anda dapat menetapkan `Sid` nilai untuk setiap pernyataan dalam array pernyataan. Jika Anda memilih untuk menggunakan `Sid` elemen, Anda harus memberikan nilai string.

Istilah terkait

- IAMJSONelemen kebijakan: Sid

Saran - Tingkatkan rentang IP

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

Menyelesaikan saran

Kondisi alamat IP harus dalam CIDR format standar, seperti 203.0.113.0/24 atau 2001:: 1234:5678: :/64. DB8 Ketika Anda memasukkan bit bukan nol setelah bit bertopeng, mereka tidak dipertimbangkan untuk kondisi tersebut. AWS merekomendasikan agar Anda menggunakan alamat baru yang disertakan dalam pesan.

- Operator kondisi alamat IP
- IAMJSONelemen kebijakan: Kondisi

Saran - Null dengan kualifikasi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

Menyelesaikan saran

Dalam `Condition` elemen, Anda membangun ekspresi di mana Anda menggunakan operator kondisi seperti sama atau kurang dari untuk membandingkan kondisi dalam kebijakan terhadap kunci dan nilai dalam konteks permintaan. Untuk permintaan yang menyertakan beberapa nilai untuk satu kunci kondisi, Anda harus menggunakan `ForAllValues` atau `ForAnyValue` mengatur operator.

Bila Anda menggunakan operator `Null` kondisi dengan`ForAllValues`, pernyataan selalu kembali`true`. Bila Anda menggunakan operator `Null` kondisi dengan`ForAnyValue`, pernyataan selalu kembali`false`. AWS merekomendasikan agar Anda menggunakan operator `StringLike` kondisi dengan operator set ini.

Istilah terkait

- Kunci konteks multivaluasi
- Operator kondisi nol
- Elemen kondisi

Saran - Subset alamat IP pribadi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

Menyelesaikan saran

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik.

Ketika `Condition` elemen Anda menyertakan campuran alamat IP pribadi dan publik, pernyataan tersebut mungkin tidak memiliki efek yang diinginkan. Anda dapat menentukan alamat IP pribadi menggunakan`aws:VpcSourceIP`.

> **Note**
>
> Kunci kondisi global hanya `aws:VpcSourceIP` cocok jika permintaan berasal dari alamat IP yang ditentukan dan melewati titik VPC akhir.

- aws: kunci kondisi SourceIp global
- aws: kunci kondisi VpcSourceIp global
- Operator kondisi alamat IP
- IAMJSONelemen kebijakan: Kondisi

Saran - Subset pribadi NotIpAddress

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

Menyelesaikan saran

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik.

Ketika `Condition` elemen Anda menyertakan operator `NotIpAddress` kondisi dan campuran alamat IP pribadi dan publik, pernyataan tersebut mungkin tidak memiliki efek yang diinginkan. Setiap alamat IP publik yang tidak ditentukan dalam kebijakan akan cocok. Tidak ada alamat IP pribadi yang cocok. Untuk mencapai efek ini, Anda dapat menggunakan `NotIpAddress` dengan `aws:VpcSourceIP` dan menentukan alamat IP pribadi yang seharusnya tidak cocok.

- aws: kunci kondisi SourceIp global
- aws: kunci kondisi VpcSourceIp global
- Operator kondisi alamat IP
- IAMJSONelemen kebijakan: Kondisi

Saran — Tindakan berlebihan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

Menyelesaikan saran

Saat Anda menggunakan wildcard (*) dalam `Action` elemen, Anda dapat menyertakan izin yang berlebihan. AWS menyarankan agar Anda meninjau kebijakan Anda dan hanya menyertakan izin yang Anda butuhkan. Ini dapat membantu Anda menghapus tindakan berlebihan.

Misalnya, tindakan berikut termasuk `iam:GetCredentialReport` tindakan dua kali.

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
    ],
```

Dalam contoh ini, izin ditentukan untuk setiap IAM tindakan yang dimulai dengan `Get` atau`List`. Saat IAM menambahkan operasi get atau list tambahan, kebijakan ini akan mengizinkannya. Anda mungkin ingin mengizinkan semua tindakan hanya-baca ini. `iam:GetCredentialReport`Tindakan ini sudah termasuk sebagai bagian dari`iam:Get*`. Untuk menghapus izin duplikat, Anda dapat menghapus. `iam:GetCredentialReport`

Anda menerima temuan untuk pemeriksaan kebijakan ini ketika semua konten tindakan berlebihan. Dalam contoh ini, jika elemen disertakan`iam:*CredentialReport`, itu tidak dianggap berlebihan. Itu termasuk`iam:GetCredentialReport`, yang berlebihan, dan`iam:GenerateCredentialReport`, yang tidak. Menghapus salah satu `iam:Get*` atau `iam:*CredentialReport` akan mengubah izin kebijakan.

- IAMJSONelemen kebijakan: Tindakan

AWS kebijakan terkelola dengan saran ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Tindakan berlebihan tidak memengaruhi izin yang diberikan oleh kebijakan. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan yang dikelola pelanggan, AWS sarankan agar Anda menghapus tindakan berlebihan dari kebijakan Anda.

Saran — Nilai kondisi redundan num

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

Menyelesaikan saran

Bila Anda menggunakan operator kondisi numerik untuk nilai serupa dalam kunci kondisi, Anda dapat membuat tumpang tindih yang menghasilkan izin berlebihan.

Misalnya, `Condition` elemen berikut mencakup beberapa `aws:MultiFactorAuthAge` kondisi yang memiliki usia tumpang tindih 1200 detik.

```
"Condition": {
        "NumericLessThan": {
          "aws:MultiFactorAuthAge": [
```

```
            "2700",
            "3600"
          ]
        }
      }
```

Dalam contoh ini, izin ditentukan jika otentikasi multi-faktor (MFA) diselesaikan kurang dari 3600 detik (1 jam) yang lalu. Anda dapat menghapus nilai redundan. `2700`

- Operator kondisi numerik
- IAMJSONelemen kebijakan: Kondisi

Saran — Sumber daya redundan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

Menyelesaikan saran

Saat menggunakan wildcard (*) di Amazon Resource Names (ARNs), Anda dapat membuat izin sumber daya yang berlebihan.

Misalnya, `Resource` elemen berikut mencakup beberapa ARNs dengan izin berlebihan.

```
"Resource": [
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

Dalam contoh ini, izin ditentukan untuk peran apa pun dengan nama yang dimulai dengan`jane`. Anda dapat menghapus `jane-admin` redundan dan `jane-s3only` ARNs tanpa mengubah izin

yang dihasilkan. Hal ini membuat kebijakan menjadi dinamis. Ini akan menentukan izin untuk setiap peran masa depan yang dimulai dengan`jane`. Jika tujuan kebijakan adalah untuk mengizinkan akses ke sejumlah peran statis, maka hapus yang terakhir ARN dan daftarkan hanya ARNs yang harus ditentukan.

- IAMJSONelemen kebijakan: Sumber daya

AWS kebijakan terkelola dengan saran ini

AWS kebijakan terkelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Sumber daya redundan tidak memengaruhi izin yang diberikan oleh kebijakan. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan yang dikelola pelanggan, AWS sarankan Anda menghapus sumber daya yang berlebihan dari kebijakan Anda.

Saran — Pernyataan berlebihan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

Menyelesaikan saran

Elemen `Statement` adalah elemen utama kebijakan ini. Elemen ini wajib diisi. Elemen `Statement` dapat berisi satu pernyataan atau serangkaian pernyataan individu.

Ketika Anda memasukkan pernyataan yang sama lebih dari sekali dalam kebijakan panjang, pernyataan tersebut berlebihan. Anda dapat menghapus salah satu pernyataan tanpa memengaruhi izin yang diberikan oleh kebijakan. Ketika seseorang mengedit kebijakan, mereka mungkin mengubah salah satu pernyataan tanpa memperbarui duplikat. Ini mungkin menghasilkan lebih banyak izin daripada yang dimaksudkan.

- IAMJSONelemen kebijakan: Pernyataan

Saran — Wildcard dalam nama layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

Menyelesaikan saran

Ketika Anda memasukkan nama AWS layanan dalam kebijakan, AWS merekomendasikan agar Anda tidak menyertakan wildcard (*,?). Ini mungkin menambahkan izin untuk layanan future yang tidak Anda inginkan. Misalnya, ada lebih dari selusin AWS layanan dengan kata `*code*` dalam nama mereka.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- IAMJSONelemen kebijakan: Sumber daya

Saran - Izinkan dengan kunci kondisi tag yang tidak didukung untuk layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
 are not allowed by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
 affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Menyelesaikan saran

Menggunakan kunci kondisi tag yang tidak didukung dalam `Condition` elemen kebijakan dengan `"Effect": "Allow"` tidak memengaruhi izin yang diberikan oleh kebijakan, karena kondisi diabaikan untuk tindakan layanan tersebut. AWS merekomendasikan agar Anda menghapus tindakan untuk layanan yang tidak mendukung kunci kondisi dan membuat pernyataan lain untuk mengizinkan akses ke sumber daya tertentu dalam layanan tersebut.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dan tidak didukung oleh tindakan layanan, maka kunci tidak termasuk dalam konteks permintaan. Dalam hal ini, kondisi dalam `Allow` pernyataan selalu kembali `false` dan tindakan tidak pernah diizinkan. Ini terjadi bahkan jika sumber daya ditandai dengan benar.

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut () ABAC. Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (). RBAC

> **Note**
>
> Beberapa layanan memungkinkan dukungan untuk kunci `aws:ResourceTag` kondisi untuk subset sumber daya dan tindakan mereka. IAMAccess Analyzer mengembalikan temuan untuk tindakan layanan yang tidak didukung. Misalnya, Amazon S3 mendukung `aws:ResourceTag` subset sumber dayanya. Untuk melihat semua jenis sumber daya yang tersedia di Amazon S3 yang mendukung kunci `aws:ResourceTag` kondisi, lihat Jenis sumber daya yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan.

Misalnya, asumsikan bahwa Anda ingin mengizinkan anggota tim untuk melihat detail sumber daya tertentu yang ditandai dengan pasangan nilai kunci. `team=BumbleBee` Juga asumsikan bahwa AWS Lambda memungkinkan Anda untuk menandai sumber daya, tetapi tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk mengizinkan tindakan tampilan untuk AWS App Mesh dan AWS Backup jika tag ini ada, gunakan kunci `aws:ResourceTag` kondisi. Untuk Lambda, gunakan

konvensi penamaan sumber daya yang menyertakan nama tim sebagai awalan. Kemudian sertakan pernyataan terpisah yang memungkinkan melihat sumber daya dengan konvensi penamaan itu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
            }
        },
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
    ]
}
```

Warning

Jangan gunakan `Not` versi operator kondisi `"Effect": "Allow"` sebagai solusi untuk temuan ini. Operator kondisi ini memberikan pencocokan yang dinegasikan. Ini berarti bahwa setelah kondisi dievaluasi, hasilnya dinegasikan. Dalam contoh sebelumnya, termasuk `lambda:GetFunction` tindakan dalam `AllowViewSupported` pernyataan dengan `StringNotEquals` operator selalu memungkinkan tindakan, terlepas dari apakah sumber daya ditandai.

Jangan gunakan... `IfExists`versi operator kondisi sebagai solusi untuk temuan ini. Ini berarti "Izinkan tindakan jika kunci ada dalam konteks permintaan dan nilainya cocok. Jika tidak, izinkan aksinya." Dalam contoh sebelumnya, termasuk `lambda:GetFunction` tindakan dalam `AllowViewSupported` pernyataan dengan `StringEqualsIfExists` operator

selalu memungkinkan tindakan. Untuk tindakan itu, kuncinya tidak ada dalam konteks, dan setiap upaya untuk melihat jenis sumber daya diizinkan, terlepas dari apakah sumber daya diberi tag.

Istilah terkait

- Kunci kondisi global
- IAMJSONelemen kebijakan: Operator kondisi
- Elemen kondisi
- Ikhtisar JSON kebijakan

Saran - Izinkan NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

Menyelesaikan saran

Menggunakan kunci kondisi tag yang tidak didukung dalam `Condition` elemen kebijakan dengan elemen `NotAction` dan `"Effect": "Allow"` tidak memengaruhi izin yang diberikan oleh kebijakan. Kondisi diabaikan untuk tindakan layanan yang tidak mendukung kunci kondisi. AWS merekomendasikan agar Anda menulis ulang logika untuk mengizinkan daftar tindakan.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dengan`NotAction`, tindakan layanan baru atau yang sudah ada yang tidak mendukung kunci tidak diperbolehkan. AWS

merekomendasikan agar Anda secara eksplisit mencantumkan tindakan yang ingin Anda izinkan. IAMAccess Analyzer mengembalikan temuan terpisah untuk tindakan terdaftar yang tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk informasi selengkapnya, lihat Saran - Izinkan dengan kunci kondisi tag yang tidak didukung untuk layanan.

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut () ABAC. Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (). RBAC

Istilah terkait

- Kunci kondisi global
- Membandingkan ABAC dengan RBAC
- IAMJSONelemen kebijakan: Operator kondisi
- Elemen kondisi
- Ikhtisar JSON kebijakan

Saran - Kunci kondisi yang direkomendasikan untuk prinsipal layanan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
 aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
 {{key}}.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

Menyelesaikan saran

Anda dapat menentukan Layanan AWS dalam `Principal` elemen kebijakan berbasis sumber daya menggunakan prinsip layanan, yang merupakan pengenal untuk layanan. Anda harus

menggunakan`aws:SourceArn`,, `aws:SourceAccount`aws:SourceOrgID, atau kunci `aws:SourceOrgPaths` kondisi saat memberikan akses ke prinsipal layanan alih-alih kunci kondisi lainnya, seperti. `aws:Referer` Ini membantu Anda mencegah masalah keamanan yang disebut masalah wakil yang bingung.

Istilah terkait

- Layanan AWS kepala sekolah
- AWS kunci kondisi global: aws: SourceAccount
- AWS kunci kondisi global: aws: SourceArn
- AWS kunci kondisi global: aws: SourceOrgId
- AWS kunci kondisi global: aws: SourceOrgPaths
- Masalah wakil yang membingungkan

Saran — Kunci kondisi yang tidak relevan dalam kebijakan

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

Menyelesaikan saran

Beberapa kunci kondisi tidak relevan untuk kebijakan berbasis sumber daya. Misalnya, kunci `s3:ResourceAccount` kondisi tidak relevan untuk kebijakan berbasis sumber daya yang dilampirkan ke bucket Amazon S3 atau jenis sumber daya jalur akses Amazon S3.

Anda harus menggunakan kunci kondisi dalam kebijakan berbasis identitas untuk mengontrol akses ke sumber daya.

Istilah terkait

- Kebijakan berbasis identitas dan kebijakan berbasis sumber daya

Saran — Prinsip redundan dalam kebijakan kepercayaan peran

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

Menyelesaikan saran

Jika Anda menentukan prinsipal peran yang diasumsikan dan peran induknya dalam `Principal` elemen kebijakan, kebijakan tersebut tidak mengizinkan atau menolak izin yang berbeda. Misalnya, itu berlebihan jika Anda menentukan `Principal` elemen menggunakan format berikut:

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

Kami merekomendasikan untuk menghapus prinsipal peran yang diasumsikan.

Istilah terkait

- Kepala sesi peran

Saran — Konfirmasi jenis klaim audiens

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Confirm audience claim type: The 'aud' (audience) claim key identifies the recipients
 that the JSON web token is intended for. Audience claims can be multivalued or single-
```

```
valued. If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If
 the claim is single-valued, do not use a qualifier.
```

Dalam panggilan terprogram ke AWS CLI or AWS API, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The 'aud' (audience) claim key identifies the recipients that the
 JSON web token is intended for. Audience claims can be multivalued or single-valued.
 If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If the claim
 is single-valued, do not use a qualifier."
```

Menyelesaikan saran

Kunci klaim `aud` (audiens) adalah pengenal unik untuk aplikasi Anda yang dikeluarkan untuk Anda saat Anda mendaftarkan aplikasi dengan iDP dan mengidentifikasi penerima yang JSON dimaksudkan untuk token web. Klaim audiens dapat bernilai multivaluasi atau bernilai tunggal. Jika klaim multivaluasi, gunakan operator set `ForAllValues` atau `ForAnyValue` kondisi. Jika klaim bernilai tunggal, jangan gunakan operator set kondisi.

Istilah terkait

- Membuat peran untuk identitas web atau OpenID Connect Federation (konsol)
- Kunci konteks multivaluasi
- Kunci kondisi bernilai tunggal vs. multivaluasi

Validasi kebijakan dengan pemeriksaan kebijakan khusus IAM Access Analyzer

Anda dapat menggunakan pemeriksaan kebijakan khusus untuk memeriksa akses baru berdasarkan standar keamanan Anda. Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Memvalidasi kebijakan dengan pemeriksaan kebijakan khusus (konsol)

Sebagai langkah opsional, Anda dapat menjalankan pemeriksaan kebijakan khusus saat mengedit kebijakan di editor JSON kebijakan di IAM konsol. Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan versi yang ada.

Untuk memeriksa akses baru saat mengedit IAM JSON kebijakan

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi di sebelah kiri, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan kebijakan yang ingin Anda edit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Izin, lalu pilih Edit.
5. Pilih JSONopsi dan buat pembaruan pada kebijakan Anda.
6. Di panel validasi kebijakan di bawah kebijakan, pilih tab Periksa akses baru, lalu pilih Periksa kebijakan. Jika izin yang dimodifikasi memberikan akses baru, pernyataan akan disorot di panel validasi kebijakan.
7. Jika Anda tidak bermaksud memberikan akses baru, perbarui pernyataan kebijakan dan pilih Periksa kebijakan hingga tidak ada akses baru yang terdeteksi.

   > **Note**
   >
   > Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

8. Pilih Berikutnya.
9. Pada halaman Tinjau dan simpan, tinjau Izin yang ditentukan dalam kebijakan ini, lalu pilih Simpan perubahan.

Memvalidasi kebijakan dengan pemeriksaan kebijakan khusus (AWS CLI atauAPI)

Anda dapat menjalankan pemeriksaan kebijakan kustom IAM Access Analyzer dari AWS CLI atau IAM Access API Analyzer.

Untuk menjalankan pemeriksaan kebijakan kustom IAM Access Analyzer ()AWS CLI

- Untuk memeriksa apakah akses baru diizinkan untuk kebijakan yang diperbarui jika dibandingkan dengan kebijakan yang ada, jalankan perintah berikut: check-no-new-access
- Untuk memeriksa apakah akses yang ditentukan tidak diizinkan oleh kebijakan, jalankan perintah berikut: check-access-not-granted

- Untuk memeriksa apakah kebijakan sumber daya dapat memberikan akses publik ke jenis sumber daya tertentu, jalankan perintah berikut: check-no-public-access

Untuk menjalankan pemeriksaan kebijakan kustom IAM Access Analyzer () API

- Untuk memeriksa apakah akses baru diizinkan untuk kebijakan yang diperbarui jika dibandingkan dengan kebijakan yang ada, gunakan CheckNoNewAccessAPIoperasi.
- Untuk memeriksa apakah akses yang ditentukan tidak diizinkan oleh kebijakan, gunakan CheckAccessNotGrantedAPIoperasi.
- Untuk memeriksa apakah kebijakan sumber daya dapat memberikan akses publik ke jenis sumber daya tertentu, gunakan CheckNoPublicAccessAPIoperasi.

IAMPembuatan kebijakan Access Analyzer

Sebagai administrator atau pengembang, Anda dapat memberikan izin kepada IAM entitas (pengguna atau peran) di luar yang mereka butuhkan. IAMmenyediakan beberapa opsi untuk membantu Anda memperbaiki izin yang Anda berikan. Salah satu opsi adalah membuat IAM kebijakan yang didasarkan pada aktivitas akses untuk suatu entitas. IAMAccess Analyzer meninjau AWS CloudTrail log Anda dan membuat templat kebijakan yang berisi izin yang digunakan entitas dalam rentang tanggal yang ditentukan. Anda dapat menggunakan templat untuk membuat kebijakan dengan izin mendetail yang memberikan hanya izin yang diperlukan untuk mendukung kasus penggunaan khusus Anda.

Topik

- Cara kerja pembuatan kebijakan
- Informasi tingkat tindakan dan layanan
- Hal yang perlu diketahui tentang pembuatan kebijakan
- Izin diperlukan untuk menghasilkan kebijakan
- Membuat kebijakan berdasarkan CloudTrail aktivitas (konsol)
- Buat kebijakan menggunakan AWS CloudTrail data di akun lain
- Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (AWS CLI)
- Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (AWS API)
- IAMLayanan pembuatan kebijakan Access Analyzer

Cara kerja pembuatan kebijakan

IAMAccess Analyzer menganalisis CloudTrail peristiwa Anda untuk mengidentifikasi tindakan dan layanan yang telah digunakan oleh IAM entitas (pengguna atau peran). Ini kemudian menghasilkan IAM kebijakan yang didasarkan pada aktivitas itu. Anda dapat menyempurnakan izin entitas saat mengganti kebijakan izin luas yang dilampirkan ke entitas dengan kebijakan yang dibuat. Berikut ini adalah gambaran umum tingkat tinggi tentang proses pembuatan kebijakan.

- Menyiapkan pembuatan templat kebijakan — Anda menentukan jangka waktu hingga 90 hari untuk IAM Access Analyzer untuk menganalisis AWS CloudTrail peristiwa historis Anda. Anda harus menentukan peran layanan yang sudah ada atau membuat peran layanan baru. Peran layanan memberikan IAM akses Access Analyzer ke CloudTrail jejak Anda dan informasi layanan yang terakhir diakses untuk mengidentifikasi layanan dan tindakan yang digunakan. Anda harus menentukan CloudTrail jejak yang mencatat peristiwa untuk akun sebelum Anda dapat membuat kebijakan. Untuk informasi selengkapnya tentang kuota IAM Access Analyzer untuk CloudTrail data, lihat kuota IAMAccess Analyzer.
- Menghasilkan kebijakan — IAM Access Analyzer menghasilkan kebijakan berdasarkan aktivitas akses di CloudTrail acara Anda.
- Meninjau dan menyesuaikan kebijakan — Setelah kebijakan dibuat, Anda dapat meninjau layanan dan tindakan yang digunakan oleh entitas selama rentang tanggal yang ditentukan. Anda dapat lebih menyesuaikan kebijakan dengan menambahkan atau menghapus izin, menentukan sumber daya, dan menambahkan syarat ke templat kebijakan.
- Buat dan lampirkan kebijakan — Anda memiliki opsi untuk menyimpan kebijakan yang dihasilkan dengan membuat kebijakan terkelola. Anda dapat melampirkan kebijakan yang Anda buat untuk pengguna atau peran yang aktivitasnya digunakan untuk menghasilkan kebijakan.

Informasi tingkat tindakan dan layanan

Saat IAM Access Analyzer membuat IAM kebijakan, informasi akan dikembalikan untuk membantu Anda menyesuaikan kebijakan lebih lanjut. Dua kategori informasi dapat dihasilkan ketika kebijakan yang membuat:

- Kebijakan dengan informasi tingkat tindakan — Untuk beberapa AWS layanan, seperti AmazonEC2, IAM Access Analyzer dapat mengidentifikasi tindakan yang ditemukan dalam CloudTrail acara Anda dan mencantumkan tindakan yang digunakan dalam kebijakan yang dihasilkannya. Untuk daftar layanan yang didukung, lihatIAMLayanan pembuatan kebijakan Access

Analyzer. Untuk beberapa layanan, IAM Access Analyzer meminta Anda untuk menambahkan tindakan untuk layanan ke kebijakan yang dihasilkan.

- Kebijakan dengan informasi tingkat layanan — IAM Access Analyzer menggunakan informasi yang terakhir diakses untuk membuat templat kebijakan dengan semua layanan yang baru digunakan. Saat menggunakan AWS Management Console, kami meminta Anda untuk meninjau layanan dan menambahkan tindakan untuk menyelesaikan kebijakan.

Untuk daftar tindakan di setiap layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan di Referensi Otorisasi Layanan.

Hal yang perlu diketahui tentang pembuatan kebijakan

Sebelum Anda membuat kebijakan, tinjau detail penting berikut ini.

- Aktifkan CloudTrail jejak — Anda harus mengaktifkan CloudTrail jejak untuk akun Anda untuk membuat kebijakan berdasarkan aktivitas akses. Saat Anda membuat CloudTrail jejak, CloudTrail kirimkan peristiwa yang terkait dengan jejak Anda ke bucket Amazon S3 yang Anda tentukan. Untuk mempelajari cara membuat CloudTrail jejak, lihat Membuat jejak untuk AWS akun Anda di Panduan AWS CloudTrail Pengguna.
- Peristiwa data tidak tersedia — IAM Access Analyzer tidak mengidentifikasi aktivitas tingkat tindakan untuk peristiwa data, seperti peristiwa data Amazon S3, dalam kebijakan yang dihasilkan.
- PassRole`iam:PassRole`Tindakan tidak dilacak oleh CloudTrail dan tidak termasuk dalam kebijakan yang dihasilkan.
- Mengurangi waktu pembuatan kebijakan — Untuk menghasilkan kebijakan lebih cepat, kurangi rentang tanggal yang Anda tentukan selama penyiapan pembuatan kebijakan.
- Gunakan CloudTrail untuk audit - Jangan gunakan pembuatan kebijakan untuk tujuan audit; gunakan CloudTrail sebagai gantinya. Untuk informasi selengkapnya tentang penggunaan CloudTrail, lihat Logging IAM dan AWS STS API panggilan dengan AWS CloudTrail.
- Tindakan yang ditolak — Pembuatan kebijakan meninjau semua CloudTrail peristiwa, termasuk tindakan yang ditolak.
- Satu IAM konsol kebijakan — Anda dapat memiliki satu kebijakan yang dihasilkan pada satu waktu di IAM konsol.
- IAMKonsol ketersediaan kebijakan yang dihasilkan — Anda dapat meninjau kebijakan yang dihasilkan di IAM konsol hingga 7 hari setelah dibuat. Setelah 7 hari, Anda harus membuat kebijakan baru.

- Kuota pembuatan kebijakan — Untuk informasi tambahan tentang kuota pembuatan kebijakan IAM Access Analyzer, lihat kuota IAMAccess Analyzer.
- Tarif standar Amazon S3 berlaku — Saat Anda menggunakan fitur pembuatan kebijakan, IAM Access Analyzer meninjau CloudTrail log di bucket S3 Anda. Tidak ada biaya penyimpanan tambahan untuk mengakses CloudTrail log Anda untuk pembuatan kebijakan. AWS mengenakan tarif standar Amazon S3 untuk permintaan dan transfer data CloudTrail log yang disimpan di bucket S3 Anda.
- AWS Control Tower dukungan — Pembuatan kebijakan tidak mendukung AWS Control Tower pembuatan kebijakan.

Izin diperlukan untuk menghasilkan kebijakan

Izin yang Anda butuhkan untuk membuat kebijakan untuk pertama kalinya berbeda dari yang Anda butuhkan untuk menghasilkan kebijakan untuk penggunaan berikutnya.

Penyiapan pertama kali

Ketika Anda membuat kebijakan untuk kali pertama, Anda harus memilih opsi Peran layanan yang sudah ada di akun Anda atau buat peran layanan baru. Peran layanan memberikan IAM akses Access Analyzer CloudTrail dan layanan informasi yang terakhir diakses di akun Anda. Hanya administrator yang harus memiliki izin yang diperlukan untuk membuat dan mengonfigurasi peran. Oleh karena itu, kami merekomendasikan agar administrator menciptakan peran layanan selama penyiapan pertama kali. Untuk mempelajari lebih lanjut tentang izin yang diperlukan untuk membuat peran layanan, lihat Membuat peran untuk mendelegasikan izin ke layanan. AWS

Izin yang diperlukan untuk peran layanan

Bila Anda membuat peran layanan, Anda mengonfigurasi dua kebijakan untuk peran tersebut. Anda melampirkan kebijakan IAM izin ke peran yang menentukan apa yang dapat dilakukan peran tersebut. Anda juga melampirkan kebijakan kepercayaan peran untuk peran yang menentukan penanggung jawab yang dapat menggunakan peran.

Contoh kebijakan pertama menunjukkan kebijakan izin untuk peran layanan yang diperlukan untuk menghasilkan kebijakan. Contoh kebijakan kedua menunjukkan kebijakan kepercayaan peran yang diperlukan untuk peran layanan. Anda dapat menggunakan kebijakan ini untuk membantu Anda membuat peran layanan saat Anda menggunakan AWS API atau AWS CLI membuat kebijakan. Saat Anda menggunakan IAM konsol untuk membuat peran layanan sebagai bagian dari proses pembuatan kebijakan, kami membuat kebijakan ini untuk Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

Contoh kebijakan berikut menunjukkan kebijakan kepercayaan peran dengan izin yang memungkinkan IAM Access Analyzer untuk mengambil peran.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
```

```
  ]
}
```

Penggunaan selanjutnya

Untuk membuat kebijakan di AWS Management Console, IAM pengguna harus memiliki kebijakan izin yang memungkinkan mereka meneruskan peran layanan yang digunakan untuk pembuatan kebijakan ke IAM Access Analyzer. `iam:PassRole`biasanya disertai dengan `iam:GetRole` agar pengguna bisa mendapatkan detail peran yang akan dilewati. Dalam contoh ini, pengguna hanya dapat meneruskan peran yang ada dalam akun yang ditentukan dengan nama yang dimulai dengan `AccessAnalyzerMonitorServiceRole*`. Untuk mempelajari selengkapnya tentang meneruskan IAM peran ke AWS layanan, lihat Memberikan izin pengguna untuk meneruskan peran AWS ke layanan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

Anda juga harus memiliki izin IAM Access Analyzer berikut untuk membuat kebijakan di AWS Management Console, AWS API, atau AWS CLI seperti yang ditunjukkan dalam pernyataan kebijakan berikut.

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
```

```
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
}
```

Untuk penggunaan pertama dan selanjutnya

Ketika Anda menggunakan AWS Management Console untuk membuat kebijakan, Anda harus memiliki `cloudtrail:ListTrails` izin untuk membuat daftar CloudTrail jejak di akun Anda seperti yang ditunjukkan dalam pernyataan kebijakan berikut.

```
{
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "CloudTrail:ListTrails"
  ],
  "Resource": "*"
}
```

Membuat kebijakan berdasarkan CloudTrail aktivitas (konsol)

Anda dapat membuat kebijakan untuk IAM pengguna atau peran.

Langkah 1: Buat kebijakan berdasarkan CloudTrail aktivitas

Prosedur berikut menjelaskan cara menghasilkan kebijakan untuk peran yang menggunakan AWS Management Console.

Menghasilkan kebijakan untuk IAM peran

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi di sebelah kiri, pilih Peran.

> **Note**
>
> Langkah-langkah untuk membuat kebijakan berdasarkan aktivitas untuk IAM pengguna hampir identik. Untuk melakukannya, pilih Pengguna sebagai ganti Peran.

3. Dalam daftar peran di akun Anda, pilih nama peran yang aktivitasnya ingin Anda gunakan untuk menghasilkan kebijakan.
4. Pada tab Izin, di bagian Buat kebijakan berdasarkan CloudTrail peristiwa, pilih Buat kebijakan.
5. Pada halaman Buat kebijakan, tentukan periode waktu yang Anda inginkan IAM Access Analyzer untuk menganalisis CloudTrail peristiwa Anda untuk tindakan yang diambil dengan peran tersebut. Anda dapat memilih rentang hingga 90 hari. Kami menyarankan Anda memilih periode waktu sesingkat mungkin untuk mengurangi waktu pembuatan kebijakan.
6. Di bagian CloudTrail akses, pilih peran yang ada yang sesuai atau buat peran baru jika peran yang sesuai tidak ada. Peran tersebut memberikan izin IAM Access Analyzer untuk mengakses CloudTrail data Anda atas nama Anda guna meninjau aktivitas akses guna mengidentifikasi layanan dan tindakan yang telah digunakan. Untuk mempelajari lebih lanjut tentang izin yang diperlukan untuk peran ini, lihatIzin diperlukan untuk menghasilkan kebijakan.
7. Di bagian CloudTrail jejak yang akan dianalisis, tentukan CloudTrail jejak yang mencatat peristiwa untuk akun tersebut.

   Jika Anda memilih CloudTrail jejak yang menyimpan log di akun yang berbeda, kotak informasi tentang akses lintas akun akan ditampilkan. Akses lintas akun memerlukan pengaturan tambahan. Untuk mempelajari lebih lanjut, lihat Choose a role for cross-account access nanti dalam topik ini.
8. Pilih Buat kebijakan.
9. Sementara pembuatan kebijakan sedang berlangsung, Anda akan dikembalikan ke halaman Ringkasan Peran tab Izin. Tunggu hingga status di bagian Detail permintaan kebijakanmenampilkan bagian Sukses, lalu pilih Lihat kebijakan yang dibuat. Anda dapat melihat kebijakan yang dihasilkan hingga tujuh hari. Jika Anda membuat kebijakan lain, kebijakan yang ada diganti dengan yang baru yang Anda buat.

Langkah 2: Tinjau izin dan tambahkan tindakan untuk layanan yang digunakan

Tinjau layanan dan tindakan yang diidentifikasi oleh IAM Access Analyzer bahwa peran yang digunakan. Anda dapat menambahkan tindakan untuk setiap layanan yang digunakan untuk templat kebijakan yang dihasilkan.

1. Tinjau bagian berikut:

- Pada Tinjau izin, tinjau daftarTindakan yang disertakan dalam kebijakan yang dihasilkan. Daftar menampilkan layanan dan tindakan yang diidentifikasi oleh IAM Access Analyzer yang digunakan oleh peran dalam rentang tanggal yang ditentukan.
- Bagian Layanan yang digunakan menampilkan layanan tambahan yang diidentifikasi oleh IAM Access Analyzer yang digunakan oleh peran dalam rentang tanggal yang ditentukan. Informasi tentang tindakan yang digunakan mungkin tidak tersedia untuk layanan yang tercantum dalam bagian ini. Gunakan menu untuk setiap layanan yang tercantum untuk secara manual memilih tindakan yang ingin Anda sertakan dalam kebijakan.

2. Setelah selesai menambahkan tindakan, pilih Next (Berikutnya).

Langkah 3: Selanjutnya sesuaikan kebijakan yang dihasilkan

Anda dapat lebih lanjut menyesuaikan kebijakan dengan menambahkan atau menghapus izin atau menentukan sumber daya.

Untuk menyesuaikan kebijakan yang dihasilkan

1. Memperbarui templat kebijakan. Templat kebijakan berisi ARN placeholder sumber daya untuk tindakan yang mendukung izin tingkat sumber daya, seperti yang ditunjukkan pada gambar berikut. Izin tingkat sumber daya mengacu pada kemampuan untuk menentukan sumber daya mana yang boleh digunakan oleh para pengguna untuk melakukan tindakan. Sebaiknya gunakan ARNsuntuk menentukan sumber daya individual dalam kebijakan untuk tindakan yang mendukung izin tingkat sumber daya. Anda dapat mengganti sumber daya placeholder ARNs dengan sumber daya yang valid ARNs untuk kasus penggunaan Anda.

   Jika suatu tindakan tidak mendukung izin tingkat sumber daya, Anda harus menggunakan wildcard (*) untuk menentukan bahwa semua sumber daya dapat dipengaruhi oleh tindakan tersebut. Untuk mempelajari AWS layanan mana yang mendukung izin tingkat sumber daya, lihat AWS layanan yang berfungsi dengannya. IAM Untuk daftar tindakan di setiap layanan, dan untuk mempelajari tindakan mana yang mendukung izin tingkat sumber daya, lihat Tindakan, Sumber Daya, dan Kunci Syarat untuk Layanan AWS.

2. (Opsional) Menambahkan, memodifikasi, atau menghapus pernyataan JSON kebijakan dalam template. Untuk mempelajari selengkapnya tentang menulis JSON kebijakan, lihat Membuat IAM kebijakan (konsol).
3. Setelah selesai menyesuaikan templat kebijakan, Anda akan memiliki opsi berikut:

   - (Opsional) Anda dapat menyalin JSON dalam template untuk digunakan secara terpisah di luar halaman Kebijakan yang dihasilkan. Misalnya, jika Anda ingin menggunakan JSON untuk membuat kebijakan di akun yang berbeda. Jika kebijakan dalam templat Anda melebihi batas 6.144 karakter untuk JSON kebijakan, kebijakan akan dibagi menjadi beberapa kebijakan.
   - Pilih Next (Berikutnya) untuk meninjau dan membuat kebijakan terkelola di akun yang sama.

Langkah 4: Tinjau dan buat kebijakan terkelola

Jika memiliki izin untuk membuat dan melampirkan IAM kebijakan, Anda dapat membuat kebijakan terkelola dari kebijakan yang dibuat. Anda kemudian dapat melampirkan kebijakan ke pengguna atau peran di akun Anda.

Untuk meninjau dan membuat kebijakan

1. Pada halaman Tinjau dan buat kebijakan terkelola, masukkan Nama dan Deskripsi (opsional) untuk kebijakan yang Anda buat.

2. (Opsional) Di bagian Summary (Ringkasan), Anda dapat meninjau izin yang akan disertakan dalam kebijakan.
3. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tag diIAM, lihat Menandai IAM sumber daya.
4. Setelah selesai, lakukan salah satu hal berikut:
   - Anda dapat melampirkan kebijakan baru langsung ke peran yang digunakan untuk menghasilkan kebijakan. Untuk melakukan ini, di dekat bagian bawah halaman, pilih kotak centang di sebelah kebijakan Lampirkan ke ***YourRoleName***. Kemudian pilih Buat dan lampirkan kebijakan.
   - Atau, pilih Buat kebijakan. Anda dapat menemukan kebijakan yang Anda buat dalam daftar kebijakan di panel navigasi Kebijakan IAM konsol.
5. Anda dapat melampirkan kebijakan yang Anda buat ke entitas di akun. Setelah Anda melampirkan kebijakan, Anda dapat menghapus kebijakan lain yang terlalu luas yang mungkin dilampirkan ke entitas. Untuk mempelajari cara melampirkan kebijakan terkelola, lihat Menambahkan izin IAM identitas (konsol).

Buat kebijakan menggunakan AWS CloudTrail data di akun lain

Anda dapat membuat CloudTrail jejak yang menyimpan data di akun pusat untuk merampingkan aktivitas yang mengatur. Misalnya, Anda dapat menggunakan AWS Organizations untuk membuat jejak yang mencatat semua peristiwa untuk semua yang ada Akun AWS di organisasi itu. Jejak itu milik akun pusat. Jika Anda ingin membuat kebijakan untuk pengguna atau peran dalam akun yang berbeda dari akun tempat data CloudTrail log Anda disimpan, Anda harus memberikan akses lintas akun. Untuk melakukannya, Anda memerlukan peran dan kebijakan bucket yang memberikan izin IAM Access Analyzer ke log Anda CloudTrail . Untuk informasi selengkapnya tentang membuat jalur Organizations, lihat Membuat jejak untuk organisasi.

Dalam contoh ini, asumsikan bahwa Anda ingin membuat kebijakan untuk pengguna atau peran di akun A. CloudTrail Jejak di akun A menyimpan CloudTrail log dalam ember di akun B. Sebelum Anda dapat membuat kebijakan, Anda harus membuat pembaruan berikut:

1. Pilih peran yang sudah ada, atau buat peran layanan baru yang memberikan IAM akses Access Analyzer ke bucket di akun B (tempat CloudTrail log Anda disimpan).

2. Verifikasi kebijakan kepemilikan objek bucket Amazon S3 dan izin bucket di akun B sehingga IAM Access Analyzer dapat mengakses objek di bucket.

Langkah 1: Pilih atau buat peran untuk akses lintas akun

- Pada layar Buat kebijakan, opsi untuk Menggunakan peran yang ada telah dipilih sebelumnya untuk Anda jika peran dengan izin yang diperlukan ada di akun Anda. Jika tidak, pilih Buat dan gunakan peran layanan baru. Peran baru digunakan untuk memberikan IAM akses Access Analyzer ke CloudTrail log Anda di akun B.

Langkah 2: Verifikasi atau perbarui konfigurasi bucket Amazon S3 Anda di akun B

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Dalam daftar Bucket, pilih nama bucket tempat log CloudTrail jejak Anda disimpan.
3. Pilih tab Permissions dan pergi ke bagian Object Ownership.

   Gunakan setelan bucket Kepemilikan Objek Amazon S3 untuk mengontrol kepemilikan objek yang Anda unggah ke bucket. Secara default, ketika objek Akun AWS upload lain ke bucket Anda, akun upload memiliki objek. Untuk membuat kebijakan, pemilik bucket harus memiliki semua objek di ember. Bergantung pada kasus ACL penggunaan, Anda mungkin perlu mengubah setelan Kepemilikan Objek untuk bucket Anda. Tetapkan Object Ownership ke salah satu opsi berikut.

   - Pemilik bucket diberlakukan (disarankan)
   - Pemilik ember lebih disukai

   > **Important**
   >
   > Agar berhasil menghasilkan kebijakan, objek dalam ember harus dimiliki oleh pemilik ember. Jika Anda memilih untuk menggunakan pemilik Bucket yang diinginkan, Anda hanya dapat membuat kebijakan untuk jangka waktu tersebut setelah perubahan kepemilikan objek dilakukan.

Untuk mempelajari lebih lanjut tentang kepemilikan objek di Amazon S3, lihat Mengontrol kepemilikan objek dan menonaktifkan bucket Anda di ACLs Panduan Pengguna Amazon S3.

4. Tambahkan izin ke kebijakan bucket Amazon S3 Anda di akun B untuk mengizinkan akses peran di akun A.

   Kebijakan contoh berikut memungkinkan `ListBucket` dan `GetObject` untuk bucket bernama`amzn-s3-demo-bucket`. Ini memungkinkan akses jika peran yang mengakses bucket milik akun di organisasi Anda dan memiliki nama yang dimulai dengan`AccessAnalyzerMonitorServiceRole`. Menggunakan `aws:PrincipalArn`sebagai `Resource` elemen `Condition` dalam memastikan bahwa peran hanya dapat mengakses aktivitas untuk akun jika itu milik akun A. Anda dapat mengganti `amzn-s3-demo-bucket` dengan nama bucket, `optional-prefix` dengan awalan opsional untuk bucket, dan `organization-id` dengan ID organisasi Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
```

```
      }
    }
  ]
}
```

5. Jika Anda mengenkripsi log Anda menggunakan AWS KMS, perbarui kebijakan AWS KMS kunci Anda di akun tempat Anda menyimpan CloudTrail log untuk memberikan IAM akses Access Analyzer untuk menggunakan kunci Anda, seperti yang ditunjukkan dalam contoh kebijakan berikut. Ganti `CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN` dengan ARN untuk jejak Anda dan `organization-id` dengan ID organisasi Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
            "s3.*.amazonaws.com"
          ],
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (AWS CLI)

Anda dapat menggunakan perintah berikut untuk menghasilkan kebijakan menggunakan AWS CLI.

Untuk membuat kebijakan

- aws accessanalyzer start-policy-generation

Untuk melihat kebijakan yang dibuat

- aws accessanalyzer get-generated-policy

Untuk membatalkan permintaan pembuatan kebijakan

- aws accessanalyzer cancel-policy-generation

Untuk melihat daftar permintaan pembuatan kebijakan

- aws accessanalyzer list-policy-generations

Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (AWS API)

Anda dapat menggunakan operasi berikut untuk membuat kebijakan menggunakan AWS API.

Untuk membuat kebijakan

- StartPolicyGeneration

Untuk melihat kebijakan yang dibuat

- GetGeneratedPolicy

Untuk membatalkan permintaan pembuatan kebijakan

- CancelPolicyGeneration

Untuk melihat daftar permintaan pembuatan kebijakan

- ListPolicyGenerations

IAMLayanan pembuatan kebijakan Access Analyzer

Tabel berikut mencantumkan AWS layanan yang IAMAccess Analyzer menghasilkan kebijakan dengan informasi tingkat tindakan. Untuk daftar tindakan di setiap layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan di Referensi Otorisasi Layanan.

Layanan	Awalan layanan
AWS Identity and Access Management Access Analyzer	akses-analyzer
AWS Account Management	akun
AWS Certificate Manager	acm
Alur Kerja Terkelola Amazon untuk Apache Airflow	aliran udara
AWS Amplify	amplify
AWS Amplify Pembuat UI	amplifyuibuilder
Amazon AppIntegrations	integrasi aplikasi
AWS AppConfig	appconfig
Amazon AppFlow	Appflow
AWS Profiler Biaya Aplikasi	application-cost-profiler
Wawasan CloudWatch Aplikasi Amazon	wawasan aplikasi
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync

Layanan	Awalan layanan
Layanan Dikelola Amazon untuk Prometheus	aps
Amazon Athena	Athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	penskalaan otomatis
AWS Marketplace	aws-marketplace
AWS Backup	pencadangan
AWS Batch	batch
Amazon Braket	braket
AWS Budgets	anggaran
AWS Cloud9	awan9
AWS CloudFormation	pembentukan awan
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartefak
AWS CodeDeploy	penyebaran kode
Amazon CodeGuru Profiler	codeguru-profiler

Layanan	Awalan layanan
CodeGuru Peninjau Amazon	pengulas codeguru
AWS CodePipeline	codepipeline
AWS CodeStar	bintang kode
AWS CodeStar Pemberitahuan	pemberitahuan bintang kode
Identitas Amazon Cognito	kognito-identitas
Kumpulan pengguna Amazon Cognito	cognito-idp
Sinkronisasi Amazon Cognito	sinkronisasi kognito
Amazon Comprehend Medical	memahami-medis
AWS Compute Optimizer	pengoptimal komputasi
AWS Config	config
Amazon Connect	hubungkan
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	pertukaran data
AWS Data Pipeline	datapipeline
Akselerator DynamoDB	dax
AWS Device Farm	devicefarm
DevOpsGuru Amazon	devops-guru
AWS Direct Connect	directconnect

Layanan	Awalan layanan
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Cluster Elastis Amazon DocumentDB	docdb-elastis
AWS Directory Service	ds
Amazon DynamoDB	dynamodb
Toko Blok Elastis Amazon	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Publik	ecr-publik
Layanan Kontainer Elastis Amazon	ecs
Layanan Amazon Elastic Kubernetes	eks
Amazon Elastic Inference	inferensi elastis
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	sistem file elastis
Elastic Load Balancing	elasticlo adbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR di EKS (EMRWadah)	kontainer emr
Amazon Tanpa EMR Server	emr-tanpa server
OpenSearch Layanan Amazon	es

Layanan	Awalan layanan
Amazon EventBridge	peristiwa
Amazon CloudWatch Terbukti	ternyata
Amazon FinSpace	ruang finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	detektor penipuan
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Gletser Amazon S3	gletser
Grafana yang Dikelola Amazon	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	stasiun tanah
Amazon GuardDuty	tugas jaga
AWS HealthLake	healthlake
Honeycode Amazon	kode madu
AWS Identity and Access Management	iam
AWS Toko Identitas	toko identitas
EC2Image Builder	imagebuilder

Layanan	Awalan layanan
Amazon Inspector Klasik	inspektur
Amazon Inspector	inspektor2
AWS IoT	iot
AWS IoT Analytics	iotanalitik
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	pembuat iottwinmaker
AWS IoT Wireless	iotwireless
Layanan Video Interaktif Amazon	ivs
Obrolan Layanan Video Interaktif Amazon	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalitik
AWS Key Management Service	km
AWS Lambda	lambda
Amazon Lex	lex

Layanan	Awalan layanan
AWS License Manager Manajer Langganan Linux	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
CloudWatch Log Amazon	log
Amazon Lookout for Equipment	peralatan pencarian
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Blockchain yang Dikelola Amazon	terkelolablockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
AWS Rekomendasi Strategi Migrasi Hub	Migration hub-strategi
Amazon Pinpoint	penargetan seluler
Amazon MQ	mq

Layanan	Awalan layanan
AWS Network Manager	manajer jaringan
Studio Amazon Nimble	gesit
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	pos terdepan
AWS Organizations	organisasi
AWS Panorama	panorama
AWS Wawasan Performa	pi
EventBridge Pipa Amazon	pipa
Amazon Polly	polly
Profil Pelanggan Amazon Connect	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Tempat Sampah Daur Ulang	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
Data Pergeseran Merah Amazon API	data pergesera n merah
AWS Migration Hub Refactor Spaces	ruang refaktor
Amazon Rekognition	rekognisi

Layanan	Awalan layanan
AWS Resilience Hub	resiliencehub
Penjelajah Sumber Daya AWS	sumber daya-penjelajah-2
AWS Resource Groups	kelompok sumber daya
AWS RoboMaker	robomaker
AWS Identity and Access Management Peran Di Mana Saja	peranandi mana saja
Rute Amazon 53	route53
Amazon Route 53 Kontrol Pemulihan	rute53- recovery-control-config
Kesiapan Pemulihan Amazon Route 53	route53-p emulihan-kesiapan
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Layanan Penyimpanan Sederhana Amazon	s3
Amazon S3 di Outposts	pos terdepan s3
Kemampuan SageMaker geospasial Amazon	pembuat sagemer-geospasial
Savings Plans	savingsplans
EventBridge Skema Amazon	skema
Amazon SimpleDB	sdb

Layanan	Awalan layanan
AWS Secrets Manager	manajer rahasia
AWS Security Hub	securityhub
Danau Keamanan Amazon	Securitylake
AWS Serverless Application Repository	repo tanpa server
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Layanan Email Amazon Sederhana	ses
AWS Shield	melindungi
AWS Signer	penandatangan
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint SMS dan Layanan Suara	sms-suara
AWS Snowball	bola salju
Layanan Antrian Sederhana Amazon	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	insiden ssm
Manajer Sistem AWS untuk SAP	ssm-sap
AWS Step Functions	negara
AWS Security Token Service	sts

Layanan	Awalan layanan
Layanan Alur Kerja Sederhana Amazon	swf
Amazon CloudWatch Synthetics	sintetis
AWS Resource Groups Tagging API	tanda
Amazon Texttract	textract
Amazon Timestream	aliran waktu
AWS Pembangun Jaringan Telekomunikasi	tnb
Amazon Transcribe	mentranskripsikan
AWS Transfer Family	transfer
Amazon Translate	menerjemahkan
ID Suara Amazon Connect	voiceid
VPCKisi Amazon	vpc-kisi
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Kebijaksanaan Amazon Connect	kebijaksanaan
Amazon WorkLink	tautan kerja
Amazon WorkSpaces	ruang kerja
AWS X-Ray	xray

IAMKuota Access Analyzer

IAMAccess Analyzer memiliki kuota berikut:

Sumber Daya	Kuota default	Kuota maksimum
Penganalisis tingkat akun maksimum per jenis penganalisis per per Wilayah Akun AWS	1	1
Penganalisis tingkat organisasi maksimum per jenis penganalisis per per Wilayah Akun AWS	5	20[1]
Aturan arsip maksimum per penganalisis	100 Setiap aturan arsip dapat memiliki hingga 20 nilai per kriteria.	1.000[1]
Jumlah maksimum pratinjau akses per penganalisis per jam	1.000	1.000
AWS CloudTrail file log diproses per generasi kebijakan	100.000	100.000
Pembuatan kebijakan yang bersamaan	1	1
Ukuran AWS CloudTrail data pembuatan kebijakan	25 GB	25 GB
Rentang AWS CloudTrail waktu pembuatan kebijakan	90 hari	90 hari
Pembuatan kebijakan per hari	Afrika (Cape Town): 5 Asia Pasifik (Hong Kong): 5 Eropa (Milan): 5	Afrika (Cape Town): 5 Asia Pasifik (Hong Kong): 5 Eropa (Milan): 5

Sumber Daya	Kuota default	Kuota maksimum
	Timur Tengah (Bahrain): 5 Semua wilayah lain yang didukung: 50 **Note** Permintaan pembuatan kebijakan yang dibatalkan berlaku untuk kuota harian.	Timur Tengah (Bahrain): 5 Semua wilayah lain yang didukung: 50

Beberapa kuota dapat dikonfigurasi pelanggan menggunakan Service Quotas.

Memecahkan masalah IAM

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah umum saat Anda bekerja dengan AWS Identity and Access Management (IAM).

Masalah

Saya tidak dapat masuk ke akun AWS saya

Verifikasi bahwa Anda memiliki kredensial yang benar dan bahwa Anda menggunakan metode yang benar untuk masuk. Untuk informasi selengkapnya, lihat Memecahkan masalah login di Panduan Pengguna.AWS Sign-In

Saya kehilangan access key saya

Access key terdiri atas dua bagian:

- Pengidentifikasi access key. Ini bukan rahasia, dan dapat dilihat di IAM konsol di mana pun kunci akses terdaftar, seperti pada halaman ringkasan pengguna.
- Secret access key. Ini tersedia saat Anda membuat pasangan access key di awal. Seperti kata sandi, kunci ini tidak dapat diambil lagi di lain waktu. Jika Anda kehilangan secret access key, Anda harus membuat pasangan access key baru. Jika Anda sudah memiliki jumlah maksimal access key, Anda harus menghapus pasangan yang ada sebelum Anda dapat membuat yang lain.

Jika Anda kehilangan secret access key, Anda harus menghapus access key dan membuat access key baru. Untuk instruksi lebih lanjut, lihatPerbarui kunci akses.

Variabel kebijakan tidak berfungsi

Jika variabel kebijakan Anda tidak berfungsi, salah satu kesalahan berikut telah terjadi:

Tanggal salah dalam elemen kebijakan Versi.

Verifikasi bahwa semua kebijakan yang mencakup variabel menyertakan nomor versi berikut dalam kebijakan: `"Version": "2012-10-17"`. Tanpa nomor versi yang benar, variabel tidak diganti selama evaluasi. Alih-alih, variabel dievaluasi secara literal. Kebijakan yang tidak menyertakan variabel masih berfungsi saat Anda menyertakan nomor versi terbaru.

Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan dibuat saat Anda mengubah kebijakan yang dikelola pelanggan diIAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola. Untuk mempelajari selengkapnya tentang elemen kebijakan `Version`, lihat IAMJSONelemen kebijakan: Version. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called “Kebijakan IAM versioning”.

Karakter variabel berada dalam kasus huruf yang salah.

Verifikasi bahwa variabel kebijakan Anda berada di kasus yang tepat. Untuk detailnya, lihat Elemen kebijakan IAM: Variabel dan tanda.

Perubahan yang saya buat tidak selalu langsung terlihat

Sebagai layanan yang diakses melalui komputer di pusat data di seluruh dunia, IAM menggunakan model komputasi terdistribusi yang disebut konsistensi akhirnya. Setiap perubahan yang Anda buat

di IAM (atau AWS layanan lain), termasuk tag access control (ABAC) berbasis atribut, membutuhkan waktu untuk terlihat dari semua titik akhir yang mungkin. Beberapa hasil penundaan dari waktu yang diperlukan untuk mengirim data dari server ke server, zona replikasi ke zona replikasi, dan Wilayah ke Wilayah. IAMjuga menggunakan caching untuk meningkatkan kinerja, tetapi dalam beberapa kasus ini dapat menambah waktu. Perubahan mungkin tidak terlihat sampai waktu data yang disimpan di-cache sebelumnya habis.

Anda harus merancang aplikasi global untuk memperhitungkan kemungkinan penundaan ini. Pastikan aplikasi bekerja sesuai harapan, bahkan ketika perubahan yang dilakukan di satu lokasi tidak secara langsung terlihat di lokasi lain. Perubahan tersebut mencakup membuat atau memperbarui pengguna, kelompok, peran, atau kebijakan. Kami menyarankan agar Anda tidak menyertakan IAM perubahan tersebut dalam jalur kode ketersediaan tinggi yang kritis dari aplikasi Anda. Sebagai gantinya, buat IAM perubahan dalam inisialisasi terpisah atau rutinitas penyiapan yang lebih jarang Anda jalankan. Selain itu, pastikan untuk memverifikasi bahwa perubahan telah dibuat merata sebelum alur kerja produksi bergantung padanya.

Untuk informasi lebih lanjut tentang bagaimana beberapa AWS layanan lain dipengaruhi oleh ini, lihat sumber daya berikut:

- Amazon DynamoDB: Baca konsistensi di Panduan Pengembang DynamoDB, dan Baca Konsistensi di Panduan Pengembang Amazon DynamoDB.
- Amazon EC2: EC2Konsistensi Akhirnya dalam EC2APIReferensi Amazon.
- Amazon EMR: Memastikan Konsistensi Saat Menggunakan Amazon S3 dan Amazon EMR untuk ETL Alur Kerja di Blog Big Data AWS
- Amazon Redshift: Mengelola Konsistensi Data dalam Panduan Pengembang Basis Data Amazon Redshift
- Amazon S3: Model Konsistensi Data Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon

Saya tidak berwenang untuk melakukan: iam: DeleteVirtual MFADevice

Anda mungkin menerima kesalahan berikut saat mencoba menetapkan atau menghapus MFA perangkat virtual untuk diri sendiri atau orang lain:

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to
 perform: iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego
 with an explicit deny
```

Ini bisa terjadi jika seseorang sebelumnya mulai menetapkan MFA perangkat virtual ke pengguna di IAM konsol dan kemudian membatalkan prosesnya. Ini menciptakan MFA perangkat virtual untuk pengguna IAM tetapi tidak pernah menetapkannya kepada pengguna. Hapus MFA perangkat virtual yang ada sebelum Anda membuat MFA perangkat virtual baru dengan nama perangkat yang sama.

Untuk mengatasi masalah ini, administrator seharusnya tidak mengedit izin kebijakan. Sebagai gantinya, administrator harus menggunakan AWS CLI atau AWS API untuk menghapus MFA perangkat virtual yang ada tetapi tidak ditetapkan.

Untuk menghapus perangkat virtual MFA yang ada tetapi belum ditetapkan

1. Lihat MFA perangkat virtual di akun Anda.
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - AWS API: `ListVirtualMFADevices`
2. Sebagai tanggapan, cari MFA perangkat virtual untuk pengguna yang Anda coba perbaiki. ARN
3. Hapus MFA perangkat virtual.
   - AWS CLI: `aws iam delete-virtual-mfa-device`
   - AWS API: `DeleteVirtualMFADevice`

Bagaimana cara membuat IAM pengguna dengan aman?

Jika Anda memiliki karyawan yang memerlukan akses AWS, Anda dapat memilih untuk membuat IAM pengguna atau menggunakan Pusat IAM Identitas untuk otentikasi. Jika Anda menggunakanIAM, AWS merekomendasikan agar Anda membuat IAM pengguna dan mengomunikasikan kredensialnya dengan aman kepada karyawan. Jika Anda tidak berada di sebelah karyawan Anda secara fisik, gunakan alur kerja yang aman untuk memberi tahu kredensialnya kepada karyawan.

Gunakan alur kerja aman berikut untuk membuat pengguna baru diIAM:

1. Buat pengguna baru menggunakan AWS Management Console. Pilih untuk memberikan AWS Management Console akses dengan kata sandi yang dihasilkan. Jika perlu, pilih kotak centang

Users must create a new password at next sign-in(Pengguna harus membuat kata sandi baru saat masuk berikutnya). Jangan tambahkan kebijakan izin ke pengguna sampai setelah mereka mengubah sandi mereka.

2. Setelah pengguna ditambahkan, salin login, nama penggunaURL, dan kata sandi untuk pengguna baru. Untuk melihat kata sandi, pilih Show (Tampilkan).
3. Kirim kata sandi ke karyawan Anda menggunakan metode komunikasi yang aman di perusahaan Anda, seperti email, obrolan, atau sistem tiket. Secara terpisah, berikan pengguna Anda tautan konsol IAM pengguna dan nama pengguna mereka. Beri tahu karyawan untuk mengonfirmasi bahwa mereka dapat berhasil masuk sebelum Anda memberikan izin kepada karyawan tersebut.
4. Setelah karyawan mengonfirmasi, tambahkan izin yang mereka perlukan. Sebagai praktik keamanan terbaik, tambahkan kebijakan yang mengharuskan pengguna untuk mengautentikasi penggunaan MFA untuk mengelola kredensialnya. Untuk contoh kebijakan, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda memecahkan masalah saat Anda bekerja dengannya. AWS

- AWS CloudTrail Panduan Pengguna — Gunakan AWS CloudTrail untuk melacak riwayat API panggilan yang dilakukan AWS dan menyimpan informasi tersebut dalam file log. Ini membantu Anda menentukan pengguna dan akun mana yang mengakses sumber daya di akun Anda, kapan panggilan dilakukan, tindakan apa yang diminta, dan banyak lagi. Untuk informasi selengkapnya, lihat Penebangan IAM dan AWS STS APIpanggilan dengan AWS CloudTrail.
- AWS Pusat Pengetahuan — Temukan FAQs dan tautkan ke sumber daya lain untuk membantu Anda memecahkan masalah.
- AWS Support Center - Dapatkan dukungan teknis.
- AWS Premium Support Center - Dapatkan dukungan teknis premium.

Memecahkan masalah akses ditolak pesan kesalahan

Informasi berikut dapat membantu Anda mengidentifikasi, mendiagnosis, dan menyelesaikan kesalahan akses yang ditolak dengan AWS Identity and Access Management. Kesalahan akses ditolak muncul saat AWS secara eksplisit atau implisit menolak permintaan otorisasi.

- Penolakan eksplisit terjadi ketika kebijakan berisi `Deny` pernyataan untuk spesifik AWS tindakan.
- Penolakan implisit terjadi ketika tidak ada pernyataan yang berlaku dan juga tidak ada `Deny` pernyataan yang berlaku`Allow`. Karena IAM kebijakan menolak IAM prinsipal secara default, kebijakan harus secara eksplisit mengizinkan prinsipal untuk melakukan suatu tindakan. Jika tidak, kebijakan tersebut secara implisit menolak akses. Untuk informasi selengkapnya, lihat Perbedaan antara penolakan tegas dan implisit..

Saat Anda mengajukan permintaan ke layanan atau sumber daya, beberapa kebijakan mungkin berlaku untuk permintaan tersebut. Tinjau semua kebijakan yang berlaku selain kebijakan yang ditentukan dalam pesan kesalahan.

- Jika beberapa kebijakan dari jenis kebijakan yang sama menolak permintaan, pesan kesalahan akses ditolak tidak menentukan jumlah kebijakan yang dievaluasi.
- Jika beberapa jenis kebijakan menolak permintaan otorisasi, AWS hanya menyertakan salah satu jenis kebijakan tersebut dalam pesan kesalahan.

> **Important**
>
> Mengalami masalah saat masuk AWS? Pastikan Anda berada di tempat yang benar AWS halaman masuk untuk jenis pengguna Anda. Jika Anda adalah Pengguna root akun AWS (pemilik akun), Anda dapat masuk AWS menggunakan kredensi yang Anda atur saat Anda membuat Akun AWS. Jika Anda seorang IAM pengguna, administrator akun Anda dapat memberi Anda AWS kredensi masuk. Jika Anda perlu meminta dukungan, jangan gunakan tautan umpan balik di halaman ini. Formulir diterima oleh AWS Tim dokumentasi, bukan AWS Support. Sebaliknya, pada halaman Hubungi Kami pilih Masih tidak dapat masuk ke halaman Anda AWS akun dan kemudian pilih salah satu opsi dukungan yang tersedia.

Saya mendapatkan “akses ditolak” ketika saya mengajukan permintaan ke AWS layanan

- Periksa apakah pesan kesalahan menyertakan jenis kebijakan yang bertanggung jawab untuk menolak akses. Misalnya, jika kesalahan menyebutkan bahwa akses ditolak karena Kebijakan Kontrol Layanan (SCP), maka Anda dapat fokus pada pemecahan masalahSCP. Setelah mengidentifikasi jenis kebijakan, Anda dapat memeriksa pernyataan penolakan atau tindakan izinkan yang hilang dalam jenis kebijakan tersebut. Jika pesan kesalahan tidak menyebutkan jenis

kebijakan yang bertanggung jawab untuk menolak akses, gunakan pedoman lainnya di bagian ini untuk memecahkan masalah lebih lanjut.

- Pastikan bahwa Anda memiliki izin kebijakan berbasis identitas untuk memanggil tindakan dan sumber daya yang Anda minta. Jika ada kondisi yang ditetapkan, Anda juga harus memenuhi kondisi tersebut saat mengirimkan permintaan. Untuk informasi tentang melihat atau memodifikasi kebijakan untuk IAM pengguna, grup, atau peran, lihatKelola IAM kebijakan.
- Jika AWS Management Console mengembalikan pesan yang menyatakan bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk bantuan. Administrator memberi Anda kredensial masuk atau tautan masuk.

  Contoh kesalahan berikut terjadi ketika `mateojackson` IAM pengguna mencoba menggunakan konsol untuk melihat detail tentang *`my-example-widget`* sumber daya fiksi tetapi tidak memiliki izin `widgets:`*`GetWidget`* fiksi.

  ```
  User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
   widgets:GetWidget on resource: my-example-widget
  ```

  Dalam hal ini, Mateo harus meminta administratornya untuk memperbarui kebijakannya untuk mengizinkan mengakses ke sumber daya *`my-example-widget`* menggunakan tindakan `widgets:`*`GetWidget`*.
- Apakah Anda mencoba mengakses layanan yang mendukung kebijakan berbasis sumber daya, seperti Amazon S3, Amazon, atau Amazon? SNS SQS Jika ya, verifikasikan bahwa kebijakan tersebut menetapkan Anda sebagai penanggung jawab dan memberi Anda akses. Jika Anda mengajukan permintaan ke layanan di dalam akun Anda, kebijakan berbasis identitas atau kebijakan berbasis sumber daya dapat memberikan izin kepada Anda. Jika Anda membuat permintaan layanan di akun yang berbeda, maka kebijakan berbasis identitas dan kebijakan berbasis sumber daya Anda harus memberi Anda izin. Untuk melihat layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM.
- Jika kebijakan Anda menyertakan kondisi dengan pasangan nilai kunci, tinjau dengan cermat. Contohnya termasuk kunci kondisi `aws:RequestTag/`*`tag-key`* global, AWS KMS `kms:EncryptionContext:`*`encryption_context_key`*, dan kunci `ResourceTag/`*`tag-key`* kondisi yang didukung oleh beberapa layanan. Pastikan bahwa nama kunci tidak cocok dengan beberapa hasil. Karena nama kunci kondisi tidak peka huruf besar/kecil, kondisi yang memeriksa kunci bernama `foo` cocok dengan `foo`, `Foo`, atau `F00`. Jika permintaan Anda menyertakan beberapa pasangan kunci-nilai dengan nama kunci yang hanya berbeda menurut

kasus, akses Anda mungkin akan ditolak secara tak terduga. Untuk informasi selengkapnya, lihat IAMJSONelemen kebijakan: Condition.

- Jika Anda memiliki batas izin, verifikasikan bahwa kebijakan yang digunakan untuk batas izin memungkinkan permintaan Anda. Jika kebijakan berbasis identitas Anda mengizinkan permintaan, tetapi batas izin Anda tidak, maka permintaan ditolak. Batas izin mengontrol izin maksimum yang dapat dimiliki IAM prinsipal (pengguna atau peran). Kebijakan berbasis sumber daya tidak terbatas oleh batasan izin. Batas izin bukanlah hal yang umum. Untuk informasi lebih lanjut tentang caranya AWS mengevaluasi kebijakan, lihatLogika evaluasi kebijakan.
- Jika Anda menandatangani permintaan secara manual (tanpa menggunakan AWS SDKs), verifikasi bahwa Anda telah menandatangani permintaan dengan benar.

Saya mendapatkan “akses ditolak” ketika saya membuat permintaan dengan kredensial keamanan sementara

- Pertama, pastikan bahwa akses Anda tidak ditolak karena alasan yang tidak terkait dengan kredensial sementara Anda. Untuk informasi selengkapnya, lihat Saya mendapatkan “akses ditolak” ketika saya mengajukan permintaan ke AWS layanan.
- Verifikasi bahwa layanan menerima kredensial keamanan sementara, lihat AWS layanan yang bekerja dengan IAM.
- Verifikasi bahwa permintaan Anda ditandatangani dengan benar dan bahwa permintaan tersebut memiliki bentuk yang baik. Untuk detailnya, lihat dokumentasi toolkit atau Gunakan kredensi sementara dengan sumber daya AWS Anda.
- Verifikasikan bahwa kredensial keamanan sementara Anda belum kedaluwarsa. Untuk informasi selengkapnya, lihat Kredensi keamanan sementara di IAM.
- Verifikasi bahwa IAM pengguna atau peran memiliki izin yang benar. Izin untuk kredenal keamanan sementara berasal dari IAM pengguna atau peran. Akibatnya, izin dibatasi untuk mereka yang diberikan peran yang kredensial sementaranya telah Anda gunakan. Untuk informasi selengkapnya tentang bagaimana izin untuk kredensial keamanan sementara ditentukan, lihat Izin untuk kredensial keamanan sementara.
- Jika Anda mengambil peran, sesi peran Anda mungkin dibatasi oleh kebijakan sesi. Saat Anda meminta kredensi keamanan sementara secara terprogram menggunakan AWS STS, Anda dapat secara opsional meneruskan kebijakan sesi inline atau terkelola. Kebijakan sesi adalah kebijakan lanjutan yang Anda sampaikan sebagai parameter saat Anda secara terprogram membuat sesi kredensial sementara untuk peran. Anda dapat meneruskan satu dokumen kebijakan sesi JSON

inline menggunakan `Policy` parameter. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas dan kebijakan sesi peran. Selain itu, jika administrator Anda atau program khusus memberi Anda kredensial sementara, mereka mungkin telah menyertakan kebijakan sesi untuk membatasi akses Anda.

- Jika Anda adalah pengguna gabungan, sesi Anda mungkin akan dibatasi oleh kebijakan sesi. Anda menjadi pengguna federasi dengan masuk ke AWS sebagai IAM pengguna dan kemudian meminta token federasi. Untuk informasi selengkapnya tentang pengguna gabungan, lihat Meminta kredensi melalui pialang identitas khusus. Jika Anda atau broker identitas Anda menyampaikan kebijakan sesi sambil meminta token federasi, sesi Anda akan dibatasi oleh kebijakan tersebut. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas IAM pengguna Anda dan kebijakan sesi. Untuk informasi selengkapnya tentang kebijakan sesi, lihat Kebijakan sesi.
- Jika Anda mengakses sumber daya yang memiliki kebijakan berbasis sumber daya dengan menggunakan peran, verifikasi bahwa kebijakan tersebut memberikan izin untuk peran tersebut. Misalnya, kebijakan berikut memungkinkan `MyRole` dari akun `111122223333` untuk mengakses `amzn-s3-demo-bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
  }]
}
```

Akses ditolak contoh pesan kesalahan

Sebagian besar pesan kesalahan akses ditolak muncul dalam format`User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*. Dalam contoh ini, *`user`* adalah Amazon Resource Name (ARN) yang tidak menerima akses, *`action`* adalah tindakan layanan yang ditolak kebijakan, dan *`resource`* adalah ARN sumber daya di mana kebijakan bertindak. Bagian *`context`* field mewakili konteks tambahan tentang jenis kebijakan yang menjelaskan mengapa kebijakan menolak akses.

Ketika kebijakan secara eksplisit menolak akses karena kebijakan berisi pernyataanDeny, maka AWS termasuk frasa `with an explicit deny in a type policy` dalam pesan kesalahan akses ditolak. Ketika kebijakan secara implisit menolak akses, maka AWS termasuk frasa `because no type policy allows the action action` dalam pesan kesalahan akses ditolak.

> **Note**
>
> Beberapa AWS layanan tidak mendukung akses ini ditolak format pesan kesalahan. Konten pesan kesalahan akses ditolak dapat bervariasi tergantung pada layanan yang membuat permintaan otorisasi.

Contoh berikut menunjukkan format untuk berbagai jenis pesan kesalahan akses ditolak.

Akses ditolak karena Kebijakan Kontrol Layanan — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam Kebijakan Kontrol Layanan Anda (SCPs). Untuk contoh berikut, tindakannya adalah`codecommit:ListRepositories`.
2. Perbarui Anda SCP dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Memperbarui SCP di AWS Organizations Panduan Pengguna.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no service control policy allows the
 codecommit:ListRespositories action
```

Akses ditolak karena Kebijakan Kontrol Layanan — penolakan eksplisit

1. Periksa `Deny` pernyataan untuk tindakan dalam Kebijakan Kontrol Layanan Anda (SCPs). Untuk contoh berikut, tindakannya adalah`codecommit:ListRepositories`.
2. Perbarui Anda SCP dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Memperbarui SCP di AWS Organizations Panduan Pengguna.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories with an explicit deny in a service control policy
```

Akses ditolak karena kebijakan VPC titik akhir — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan titik akhir Virtual Private Cloud (VPC) Anda. Untuk contoh berikut, tindakannya adalah`codecommit:ListRepositories`.
2. Perbarui kebijakan VPC titik akhir Anda dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Memperbarui kebijakan VPC titik akhir di AWS PrivateLink Panduan.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no VPC endpoint policy allows the
 codecommit:ListRepositories action
```

Akses ditolak karena kebijakan VPC titik akhir — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan titik akhir Virtual Private Cloud (VPC) Anda. Untuk contoh berikut, tindakannya adalah`codedeploy:ListDeployments`.
2. Perbarui kebijakan VPC titik akhir Anda dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Memperbarui kebijakan VPC titik akhir di AWS PrivateLink Panduan.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a VPC endpoint policy
```

Akses ditolak karena batas izin — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan di batas izin Anda. Untuk contoh berikut, tindakannya adalah`codedeploy:ListDeployments`.
2. Perbarui batas izin Anda dengan menambahkan `Allow` pernyataan ke kebijakan Anda. IAM Untuk informasi selengkapnya, silakan lihat Batas izin untuk entitas IAM dan Edit IAM kebijakan.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* because no permissions boundary allows the
 codedeploy:ListDeployments action
```

Akses ditolak karena batas izin — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan di batas izin Anda. Untuk contoh berikut, tindakannya adalah`sagemaker:ListModels`.
2. Perbarui batas izin Anda dengan menghapus `Deny` pernyataan dari kebijakan Anda. IAM Untuk informasi selengkapnya, silakan lihat Batas izin untuk entitas IAM dan Edit IAM kebijakan.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sagemaker:ListModels with an explicit deny in a permissions boundary
```

Akses ditolak karena kebijakan sesi — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan sesi Anda. Untuk contoh berikut, tindakannya adalah`codecommit:ListRepositories`.
2. Perbarui kebijakan sesi Anda dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan sesi danEdit IAM kebijakan.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no session policy allows the
 codecommit:ListRepositories action
```

Akses ditolak karena kebijakan sesi — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan sesi Anda. Untuk contoh berikut, tindakannya adalah`codedeploy:ListDeployments`.
2. Perbarui kebijakan sesi Anda dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan sesi danEdit IAM kebijakan.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a sessions policy
```

Akses ditolak karena kebijakan berbasis sumber daya — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan berbasis sumber daya Anda. Untuk contoh berikut, tindakannya adalah`secretsmanager:GetSecretValue`.
2. Perbarui kebijakan Anda dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan berbasis sumber daya dan. Edit IAM kebijakan

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
secretsmanager:GetSecretValue because no resource-based policy allows the
 secretsmanager:GetSecretValue action
```

Akses ditolak karena kebijakan berbasis sumber daya — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan berbasis sumber daya Anda. Untuk contoh berikut, tindakannya adalah`secretsmanager:GetSecretValue`.
2. Perbarui kebijakan Anda dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan berbasis sumber daya dan. Edit IAM kebijakan

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
secretsmanager:GetSecretValue on resource: arn:aws:secretsmanager:us-
east-1:123456789012:secret:* with an explicit deny in a resource-based policy
```

Akses ditolak karena kebijakan kepercayaan peran — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan kepercayaan peran Anda. Untuk contoh berikut, tindakannya adalah`sts:AssumeRole`.
2. Perbarui kebijakan Anda dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan berbasis sumber daya dan. Edit IAM kebijakan

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
sts:AssumeRole because no role trust policy allows the sts:AssumeRole action
```

Akses ditolak karena kebijakan kepercayaan peran — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan kepercayaan peran Anda. Untuk contoh berikut, tindakannya adalah`sts:AssumeRole`.
2. Perbarui kebijakan Anda dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan berbasis sumber daya dan. Edit IAM kebijakan

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sts:AssumeRole with an explicit deny in the role trust policy
```

Akses ditolak karena kebijakan berbasis identitas — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan berbasis identitas yang dilampirkan pada identitas. Untuk contoh berikut, tindakan `codecommit:ListRepositories` dilampirkan pada peranHR.
2. Perbarui kebijakan Anda dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan berbasis identitas dan. Edit IAM kebijakan

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
codecommit:ListRepositories because no identity-based policy allows the
 codecommit:ListRepositories action
```

Akses ditolak karena kebijakan berbasis identitas — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan berbasis identitas yang dilampirkan pada identitas. Untuk contoh berikut, tindakan `codedeploy:ListDeployments` dilampirkan pada peranHR.
2. Perbarui kebijakan Anda dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan berbasis identitas dan. Edit IAM kebijakan

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in an identity-based policy
```

Akses ditolak ketika VPC permintaan gagal karena kebijakan lain

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam Kebijakan Kontrol Layanan Anda (SCPs). Untuk contoh berikut, tindakannya adalah`SNS:Publish`.
2. Perbarui Anda SCP dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Memperbarui SCP di AWS IAM Identity Center Panduan Pengguna.

```
User: arn:aws:sts::111122223333:assumed-role/role-name/role-session-name is not
 authorized to perform:
SNS:Publish on resource: arn:aws:sns:us-east-1:444455556666:role-name-2
with an explicit deny in a VPC endpoint policy transitively through a service control
 policy
```

Memecahkan masalah dengan pengguna root

Gunakan informasi di sini untuk membantu Anda memecahkan masalah yang terkait dengan pengguna root file. Akun AWS

Saya tidak dapat melakukan tugas yang saya harapkan dapat dilakukan saat masuk sebagai pengguna root akun

Akun Anda mungkin anggota organisasi di AWS Organizations. Administrator organisasi Anda mungkin memiliki kebijakan kontrol layanan (SCP) untuk membatasi izin akun Anda. SCPsberdampak pada semua pengguna, termasuk pengguna root. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan di Panduan Pengguna AWS Organizations .

Saya lupa kata sandi pengguna root untuk saya Akun AWS

Jika Anda adalah pengguna root dan Anda telah kehilangan atau lupa kata sandi untuk Anda Akun AWS, Anda dapat mengatur ulang kata sandi Anda. Anda harus mengetahui alamat email yang digunakan untuk membuat Akun AWS, dan Anda harus memiliki akses ke akun email. Untuk informasi selengkapnya, lihat Setel ulang kata sandi pengguna root yang hilang atau terlupakan.

Saya tidak memiliki akses ke email untuk saya Akun AWS

Saat Anda membuat Akun AWS, Anda memberikan alamat email dan kata sandi. Ini adalah kredensial untuk Pengguna root akun AWS. Jika Anda tidak yakin dengan alamat email yang terkait

dengan Anda Akun AWS, cari pesan yang dikirim dari `@signin.aws` atau `@verify.signin.aws` ke alamat email untuk organisasi Anda yang mungkin telah digunakan untuk membuka Akun AWS.

Jika Anda tahu alamat email tetapi tidak lagi memiliki akses ke email, cobalah untuk memulihkan akses ke email. Gunakan salah satu opsi berikut untuk mendapatkan kembali akses ke email Anda:

- Jika Anda memiliki domain untuk alamat email, Anda dapat mengembalikan alamat email yang dihapus. Atau, Anda dapat mengatur tangkapan semua untuk akun email Anda. Catch-all mengumpulkan semua pesan yang dikirim ke alamat email yang tidak lagi ada di server email dan mengarahkannya ke alamat email lain.
- Jika alamat email pada akun adalah bagian dari sistem email perusahaan Anda, kami sarankan untuk menghubungi administrator sistem IT Anda. Mereka mungkin dapat membantu Anda mendapatkan akses ke email tersebut.

Jika Anda masih tidak dapat masuk Akun AWS, Anda dapat menemukan opsi dukungan alternatif di Hubungi kami.

Memecahkan masalah kebijakan IAM

Kebijakan adalah entitas di AWS bahwa, ketika dilampirkan ke identitas atau sumber daya, mendefinisikan izin mereka. AWS mengevaluasi kebijakan ini ketika prinsipal, seperti pengguna, membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Kebijakan disimpan di AWS sebagai JSON dokumen yang dilampirkan pada prinsipal sebagai kebijakan berbasis identitas atau sumber daya sebagai kebijakan berbasis sumber daya. Anda dapat melampirkan kebijakan berbasis identitas ke prinsipal (atau identitas), seperti IAM grup, pengguna, atau peran. Kebijakan berbasis identitas meliputi AWS kebijakan terkelola, kebijakan yang dikelola pelanggan, dan kebijakan inline. Anda dapat membuat dan mengedit kebijakan yang dikelola pelanggan di AWS Management Console menggunakan opsi Visual dan JSONeditor. Saat Anda melihat kebijakan di AWS Management Console, Anda dapat melihat ringkasan izin yang diberikan oleh kebijakan tersebut. Anda dapat menggunakan editor visual dan ringkasan kebijakan untuk membantu Anda mendiagnosis dan memperbaiki kesalahan umum yang dihadapi saat mengelola IAM kebijakan.

Perlu diingat bahwa semua IAM kebijakan disimpan menggunakan sintaks yang dimulai dengan aturan JavaScript Object Notation ()JSON. Anda tidak perlu memahami sintaksis ini untuk membuat atau mengelola kebijakan Anda. Anda dapat membuat dan mengedit kebijakan menggunakan editor

visual di AWS Management Console. Untuk mempelajari lebih lanjut tentang JSON sintaks dalam IAM kebijakan, lihatTata bahasa IAM JSON kebijakan .

Topik Kebijakan Pemecahan Masalah IAM

- Atasi masalah menggunakan editor visual
  - Restrukturisasi kebijakan
  - Memilih sumber daya ARN di editor visual
  - Menolak izin di editor visual
  - Menentukan beberapa layanan di editor visual
  - Mengurangi ukuran kebijakan Anda di editor visual
  - Perbaikan layanan, tindakan, atau tipe sumber daya yang tidak dikenal di editor visual
- Memecahkan masalah dengan ringkasan kebijakan
  - Ringkasan kebijakan hilang
  - Ringkasan kebijakan mencakup layanan, tindakan, atau tipe sumber daya yang tidak dikenal
  - Layanan tidak mendukung ringkasan IAM kebijakan
  - Kebijakan saya tidak memberikan izin yang diharapkan
- Pemecahan masalah manajemen kebijakan
  - Melampirkan atau melepaskan kebijakan di akun IAM
  - Mengubah kebijakan IAM identitas Anda berdasarkan aktivitasnya
- Memecahkan masalah dokumen kebijakan JSON
  - Memvalidasi kebijakan Anda
  - Saya tidak memiliki izin untuk validasi kebijakan di editor JSON
  - Lebih dari satu objek JSON kebijakan
  - Lebih dari satu elemen JSON pernyataan
  - Lebih dari satu efek, tindakan, atau elemen sumber daya dalam elemen JSON pernyataan
  - Elemen JSON versi yang hilang

Atasi masalah menggunakan editor visual

Saat membuat atau mengedit kebijakan yang dikelola pelanggan, Anda dapat menggunakan informasi di editor Visual untuk membantu memecahkan masalah kesalahan dalam kebijakan Anda.

Untuk melihat contoh penggunaan editor visual untuk membuat kebijakan, lihat the section called “Mengontrol akses ke identitas”.

Restrukturisasi kebijakan

Saat Anda membuat kebijakan, AWS memvalidasi, memproses, dan mengubah kebijakan sebelum menyimpannya. Ketika kebijakan diambil, AWS mengubahnya kembali ke format yang dapat dibaca manusia tanpa mengubah izin. Hal ini dapat menyebabkan perbedaan dalam apa yang Anda lihat di editor visual kebijakan atau JSONtab.

- Blok izin editor visual dapat ditambahkan, dihapus, atau disusun ulang, dan konten di dalam blok dapat dioptimalkan.
- Di JSONtab, ruang putih yang tidak signifikan dapat dihapus, dan elemen dalam JSON peta dapat disusun ulang. Selain itu, Akun AWS IDsdalam elemen utama dapat diganti dengan Amazon Resource Name (ARN) dari Pengguna root akun AWS.

Karena kemungkinan perubahan ini, Anda tidak boleh membandingkan dokumen JSON kebijakan sebagai string.

Saat Anda membuat kebijakan terkelola pelanggan di AWS Management Console, Anda dapat memilih untuk bekerja sepenuhnya di JSONeditor. Jika Anda tidak pernah mengubah kebijakan di editor Visual dan memilih Berikutnya dari JSONeditor, kebijakan tersebut cenderung tidak direstrukturisasi. Bila Anda menggunakan editor Visual, IAM mungkin akan merestrukturisasi kebijakan untuk mengoptimalkan tampilannya. Restrukturisasi ini hanya ada pada sesi pengeditan Anda dan tidak disimpan secara otomatis.

Jika kebijakan Anda direstrukturisasi dalam sesi pengeditan, IAM tentukan apakah akan menyimpan restrukturisasi berdasarkan situasi berikut:

Menggunakan opsi editor ini	Jika Anda mengedit kebijakan	Dan kemudian pilih Berikutnya dari tab ini	Saat Anda memilih Simpan perubahan
Visual	Diedit	Visual	Kebijakan distruktur ulang
Visual	Diedit	JSON	Kebijakan distruktur ulang

Menggunakan opsi editor ini	Jika Anda mengedit kebijakan	Dan kemudian pilih Berikutnya dari tab ini	Saat Anda memilih Simpan perubahan
Visual	Tidak diedit	Visual	Kebijakan distruktur ulang
JSON	Diedit	Visual	Kebijakan distruktur ulang
JSON	Diedit	JSON	Struktur kebijakan tidak diubah
JSON	Tidak diedit	JSON	Struktur kebijakan tidak diubah

IAMmungkin merestrukturisasi kebijakan atau kebijakan kompleks yang memiliki blok izin atau pernyataan yang memungkinkan beberapa layanan, jenis sumber daya, atau kunci kondisi.

Memilih sumber daya ARN di editor visual

Saat Anda membuat atau mengedit kebijakan menggunakan editor visual, Anda harus memilih layanan terlebih dahulu, kemudian memilih tindakan dari layanan tersebut. Jika layanan dan tindakan yang Anda pilih mendukung pemilihan sumber daya tertentu, editor visual mencantumkan tipe sumber daya yang didukung. Anda kemudian dapat memilih Tambah ARN untuk memberikan detail tentang sumber daya Anda. Anda dapat memilih dari opsi berikut untuk menambahkan ARN untuk jenis sumber daya.

- Gunakan ARN pembangun — Anda mungkin melihat bidang yang berbeda untuk membangun ARN berdasarkan jenis sumber daya. Anda juga dapat memilih Apa pun untuk memberikan izin bagi nilai apa pun untuk pengaturan yang ditentukan. Misalnya, jika Anda memilih grup tingkat akses Amazon EC2 Baca, tindakan dalam kebijakan Anda mendukung jenis `instance` sumber daya. Berikan Wilayah, Akun, dan InstanceIdnilai untuk sumber daya Anda. Kebijakan ini memberikan izin ke instans apa pun di akun Anda jika Anda memberikan ID akun dan memilih Any for the Region dan ID instans.
- Ketik atau tempel ARN - Anda dapat menentukan sumber daya berdasarkan Nama Sumber Daya Amazon mereka (ARN). Anda dapat menyertakan karakter wildcard (*****) di bidang apa pun dari ARN (antara setiap pasangan titik dua). Untuk informasi selengkapnya, lihat IAMJSONelemen kebijakan: Resource.

Menolak izin di editor visual

Secara default, kebijakan yang Anda buat menggunakan editor visual memungkinkan tindakan yang Anda pilih. Untuk menolak tindakan yang dipilih, pilih Beralih ke menolak izin. Karena permintaan ditolak secara default, sebaiknya Anda mengizinkan izin hanya untuk tindakan dan sumber daya yang dibutuhkan pengguna. Anda harus membuat pernyataan penolakan hanya jika Anda ingin mengganti izin secara terpisah yang diizinkan oleh pernyataan atau kebijakan lain. Kami sarankan Anda membatasi jumlah izin penolakan seminim mungkin karena dapat meningkatkan kesulitan izin pemecahan masalah. Untuk informasi selengkapnya tentang cara IAM mengevaluasi logika kebijakan, lihatLogika evaluasi kebijakan.

> **Note**
>
> Secara default, hanya Pengguna root akun AWS memiliki akses ke semua sumber daya di akun itu. Jadi, jika Anda belum masuk sebagai pengguna utama, Anda harus memiliki izin yang diberikan oleh kebijakan.

Menentukan beberapa layanan di editor visual

Saat Anda menggunakan editor visual untuk membuat kebijakan, Anda hanya dapat memilih satu layanan pada satu waktu. Ini adalah praktik terbaik karena editor visual kemudian mengizinkankan Anda memilih dari tindakan untuk satu layanan tersebut. Kemudian Anda memilih dari sumber daya yang didukung oleh layanan tersebut dan tindakan yang dipilih. Ini mempermudah pembuatan dan pemecahan masalah kebijakan Anda.

Anda juga dapat menggunakan karakter wildcard (*) untuk menentukan beberapa layanan secara manual. Sebagai contoh, ketik **`Code*`** untuk memberikan izin untuk semua layanan yang dimulai dengan `Code`, seperti `CodeBuild` dan `CodeCommit`. Namun, Anda kemudian harus mengetik tindakan dan sumber daya ARNs untuk menyelesaikan kebijakan Anda. Sebagai tambahan, saat Anda menyimpan kebijakan Anda, kebijakan itu mungkin distruktur ulang untuk menyertakan setiap layanan di blok izin terpisah.

Atau, untuk menggunakan JSON sintaks (seperti wildcard) untuk layanan, buat, edit, dan simpan kebijakan Anda menggunakan opsi JSONeditor.

Mengurangi ukuran kebijakan Anda di editor visual

Saat Anda menggunakan editor visual untuk membuat kebijakan, IAM buat JSON dokumen untuk menyimpan kebijakan Anda. Anda dapat melihat dokumen ini dengan beralih ke opsi JSONeditor. Jika JSON dokumen ini melebihi batas ukuran kebijakan, editor visual akan menampilkan pesan kesalahan. Anda tidak akan dapat meninjau dan menyimpan kebijakan. Untuk melihat IAM batasan ukuran kebijakan terkelola, lihatIAMdan batas STS karakter.

Untuk mengurangi ukuran kebijakan Anda di editor visual, edit kebijakan Anda atau pindahkan blok izin ke kebijakan lain. Pesan galat menyertakan jumlah karakter yang berisi dokumen kebijakan Anda. Anda dapat menggunakan informasi ini untuk membantu Anda mengurangi ukuran kebijakan Anda.

Perbaikan layanan, tindakan, atau tipe sumber daya yang tidak dikenal di editor visual

Anda mungkin melihat peringatan di editor visual bahwa kebijakan Anda menyertakan jenis layanan, tindakan, atau sumber daya yang tidak dikenal.

> **Note**
>
> IAMmeninjau nama layanan, tindakan, dan jenis sumber daya untuk layanan yang mendukung ringkasan kebijakan. Akan tetapi, ringkasan kebijakan Anda mungkin mencakup nilai sumber daya atau kondisi yang tidak ada. Selalu uji kebijakan Anda dengan simulator kebijakan.

Jika kebijakan Anda mencakup layanan, tindakan, atau jenis sumber daya yang tidak dikenal, salah satu kesalahan berikut telah terjadi:

- Layanan pratinjau – Layanan yang ada di pratinjau tidak mendukung editor visual. Jika Anda berpartisipasi dalam pratinjau, Anda harus mengetik tindakan dan sumber daya secara manual ARNs untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi JSONeditor untuk mengetik atau menempelkan dokumen JSON kebijakan.
- Layanan khusus – Layanan khusus tidak mendukung editor visual. Jika Anda menggunakan layanan kustom, Anda harus mengetik tindakan dan sumber daya secara manual ARNs untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi JSONeditor untuk mengetik atau menempelkan dokumen JSON kebijakan.

- Layanan tidak mendukung editor visual — Jika kebijakan Anda menyertakan layanan yang tersedia secara umum (GA) yang tidak mendukung editor visual, Anda harus mengetik tindakan dan sumber daya secara manual ARNs untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi JSONeditor untuk mengetik atau menempelkan dokumen JSON kebijakan.

  Layanan yang tersedia secara umum adalah layanan yang dirilis secara publik dan bukan pratinjau atau layanan khusus. Jika layanan yang tidak dikenal tersedia secara umum dan nama dieja dengan benar, maka layanan tersebut tidak mendukung editor visual. Untuk mempelajari cara meminta dukungan editor visual atau ringkasan kebijakan untuk layanan GA, lihat Layanan tidak mendukung ringkasan IAM kebijakan.
- Tindakan tidak mendukung editor visual — Jika kebijakan Anda menyertakan layanan yang didukung dengan tindakan yang tidak didukung, Anda harus mengetik tindakan dan sumber daya secara manual ARNs untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi JSONeditor untuk mengetik atau menempelkan dokumen JSON kebijakan.

  Jika kebijakan Anda menyertakan layanan yang didukung dengan tindakan yang tidak didukung, maka layanan tersebut tidak sepenuhnya mendukung editor visual. Untuk mempelajari cara meminta dukungan editor visual atau ringkasan kebijakan untuk layanan GA, lihat Layanan tidak mendukung ringkasan IAM kebijakan.
- Jenis sumber daya tidak mendukung editor visual – Jika kebijakan Anda mencakup tindakan yang didukung dengan jenis sumber daya yang tidak didukung, Anda dapat mengabaikan peringatan dan melanjutkan. Namun, IAM tidak dapat mengonfirmasi bahwa Anda telah menyertakan sumber daya untuk semua tindakan yang dipilih, dan Anda mungkin melihat peringatan tambahan.
- Salah ketik – Saat Anda mengetikkan layanan, tindakan, atau sumber daya secara manual dalam editor visual, Anda dapat membuat kebijakan yang mencakup kesalahan ketik. Kami menyarankan Anda menggunakan editor visual dengan memilih dari daftar layanan dan tindakan. Kemudian, lengkapi bagian sumber daya sesuai dengan petunjuknya. Jika layanan tidak sepenuhnya mendukung editor visual, Anda mungkin harus mengetik bagian kebijakan secara manual.

  Jika Anda yakin bahwa kebijakan Anda tidak berisi satu pun kesalahan di atas, maka kebijakan Anda mungkin mengandung kesalahan ketik. Periksa masalah berikut:

  - Nama layanan, tindakan, dan jenis sumber daya yang salah eja, seperti `s2` alih-alih `s3` atau `ListMyBuckets` bukan `ListAllMyBuckets`
  - Teks yang tidak perluARNs, seperti `arn:aws:s3: : :*`

- Hilang titik dua dalam tindakan, seperti `iam.CreateUser`

Anda dapat mengevaluasi kebijakan yang mungkin menyertakan kesalahan ketik dengan memilih Berikutnya untuk meninjau ringkasan kebijakan. Kemudian, konfirmasikan apakah kebijakan tersebut memberikan izin yang Anda inginkan.

Memecahkan masalah dengan ringkasan kebijakan

Anda dapat mendiagnosis dan menyelesaikan masalah terkait rangkuman kebijakan.

Ringkasan kebijakan hilang

IAMKonsol menyertakan tabel ringkasan kebijakan yang menjelaskan tingkat akses, sumber daya, dan kondisi yang diizinkan atau ditolak untuk setiap layanan dalam kebijakan. Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan kebijakan mencakup daftar layanan dan ringkasan izin yang ditentukan oleh kebijakan yang dipilih. Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke entitas di halaman Detail Kebijakan untuk kebijakan tersebut. Anda dapat melihat ringkasan kebijakan untuk kebijakan yang dikelola pada halaman Kebijakan. Jika AWS tidak dapat merender ringkasan untuk kebijakan, Anda akan melihat dokumen JSON kebijakan dan kesalahan berikut:

Ringkasan untuk kebijakan ini tidak dapat dibuat. Anda masih dapat melihat atau mengedit dokumen JSON kebijakan.

Jika kebijakan Anda tidak mencakup ringkasan, salah satu kesalahan berikut telah terjadi:

- Elemen kebijakan yang tidak didukung - IAM tidak mendukung pembuatan ringkasan kebijakan untuk kebijakan yang mencakup salah satu elemen kebijakan berikut:
  - `Principal`
  - `NotPrincipal`
  - `NotResource`
- Tidak ada izin kebijakan – Jika kebijakan tidak menyediakan izin yang efektif, maka ringkasan kebijakan tidak dapat dibuat. Misalnya, jika kebijakan mencakup satu pernyataan dengan elemen `"NotAction": "*"`, maka akses ke semua tindakan akan diberikan kecuali “semua tindakan” (*). Ini berarti tidak ada akses `Deny` atau `Allow` yang diberikan.

> **Note**
>
> Hati-hati saat menggunakan elemen kebijakan ini seperti`NotPrincipal`,`NotAction`, dan`NotResource`. Untuk informasi selengkapnya tentang elemen-elemen kebijakan, lihat IAMJSONreferensi elemen kebijakan.

Jika Anda menyediakan layanan dan sumber daya yang tidak cocok, Anda dapat membuat kebijakan yang tidak memberikan izin yang efektif. Ini dapat terjadi ketika Anda menentukan tindakan dalam satu layanan dan sumber daya dari layanan lain. Dalam hal ini, ringkasan kebijakan muncul. Satu-satunya indikasi bahwa terdapat masalah adalah kolom sumber daya di ringkasan dapat mencakup sumber daya dari layanan yang berbeda. Jika kolom ini memuat sumber daya yang tidak sesuai, maka Anda harus meninjau kebijakan Anda untuk kesalahan. Uji kebijakan Anda dengan simulator kebijakan untuk lebih memahami kebijakan.

Ringkasan kebijakan mencakup layanan, tindakan, atau tipe sumber daya yang tidak dikenal

Di IAM konsol, jika ringkasan kebijakan menyertakan simbol peringatan
( ),
kebijakan tersebut mungkin menyertakan jenis layanan, tindakan, atau sumber daya yang tidak dikenal. Untuk mempelajari tentang peringatan di dalam ringkasan kebijakan, lihat Ringkasan kebijakan (daftar layanan).

> **Note**
>
> IAMmeninjau nama layanan, tindakan, dan jenis sumber daya untuk layanan yang mendukung ringkasan kebijakan. Akan tetapi, ringkasan kebijakan Anda mungkin mencakup nilai sumber daya atau kondisi yang tidak ada. Selalu uji kebijakan Anda dengan simulator kebijakan.

Jika kebijakan Anda mencakup layanan, tindakan, atau jenis sumber daya yang tidak dikenal, salah satu kesalahan berikut telah terjadi:

- Layanan pratinjau – Layanan yang ada di pratinjau tidak mendukung rangkuman kebijakan.

- Layanan khusus – Layanan khusus tidak mendukung rangkuman kebijakan.
- Layanan tidak mendukung rangkuman – Jika kebijakan Anda mencakup layanan yang tersedia secara umum (GA) yang tidak mendukung rangkuman kebijakan, maka layanan tersebut termasuk dalam bagian Layanan yang tidak dikenal di tabel ringkasan kebijakan. Layanan yang tersedia secara umum adalah layanan yang dirilis secara publik dan bukan pratinjau atau layanan khusus. Jika layanan yang tidak dikenal umumnya tersedia dan nama dieja dengan benar, maka layanan tidak mendukung ringkasan IAM kebijakan. Untuk mempelajari cara meminta dukungan ringkasan kebijakan untuk layanan GA, lihat Layanan tidak mendukung ringkasan IAM kebijakan.
- Tindakan tidak mendukung rangkuman – Jika kebijakan Anda mencakup layanan yang didukung dengan tindakan yang tidak didukung, maka tindakan tersebut termasuk dalam bagian Tindakan yang tidak dikenal pada tabel ringkasan layanan. Untuk mempelajari tentang peringatan dalam rangkuman layanan, lihat Ringkasan layanan (daftar tindakan).
- Jenis sumber daya tidak mendukung rangkuman – Jika kebijakan Anda mencakup tindakan yang didukung dengan jenis sumber daya yang tidak didukung, maka sumber daya tersebut disertakan dalam bagian Jenis sumber daya yang tidak dikenal pada tabel ringkasan layanan. Untuk mempelajari tentang peringatan di dalam rangkuman layanan, lihat Ringkasan layanan (daftar tindakan).
- Salah ketik — AWS memeriksa bahwa kebijakan tersebut JSON benar secara sintaksis, dan kebijakan tersebut tidak menyertakan kesalahan ketik atau kesalahan lain sebagai bagian dari validasi kebijakan.

> **Note**
>
> Sebagai praktik terbaik, kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi IAM kebijakan Anda guna memastikan izin yang aman dan fungsional. Kami menyarankan Anda membuka kebijakan yang ada dan meninjau serta menyelesaikan rekomendasi validasi kebijakan apa pun.

Layanan tidak mendukung ringkasan IAM kebijakan

Dimungkinkan untuk ringkasan IAM kebijakan atau editor visual untuk tidak mendukung layanan atau tindakan yang tersedia secara umum (GA). Layanan yang tersedia secara umum adalah layanan yang dirilis secara publik dan bukan layanan yang dipratinjau atau khusus. Jika layanan yang tidak dikenal tersedia secara umum dan nama dieja dengan benar, maka layanan tidak mendukung

fitur-fitur ini. Jika kebijakan Anda menyertakan layanan yang didukung dengan tindakan yang tidak didukung, maka layanan tidak sepenuhnya mendukung ringkasan IAM kebijakan.

Untuk meminta layanan menambahkan ringkasan IAM kebijakan atau dukungan editor visual

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Temukan kebijakan yang mencakup layanan yang tidak didukung:
   - Jika kebijakan tersebut adalah kebijakan terkelola, pilih Kebijakan di panel navigasi. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
   - Jika kebijakan tersebut adalah kebijakan inline yang terlampir pada pengguna, pilih Pengguna di panel navigasi. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat. Dalam tabel kebijakan untuk pengguna, perluas kop untuk ringkasan kebijakan yang ingin Anda lihat.
3. Di sisi kiri di AWS Management Console footer, pilih Umpan Balik. Di IAM kotak Umpan Balik untuk, ketik**`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`**. Jika Anda menginginkan lebih dari satu layanan untuk mendukung ringkasan, ketik **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**.

Untuk meminta layanan menambahkan dukungan ringkasan IAM kebijakan untuk tindakan yang hilang

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Temukan kebijakan yang mencakup layanan yang tidak didukung:
   - Jika kebijakan tersebut adalah kebijakan terkelola, pilih Kebijakan di panel navigasi. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
   - Jika kebijakan tersebut adalah kebijakan inline yang terlampir pada pengguna, pilih Pengguna di panel navigasi. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat. Dalam tabel kebijakan untuk pengguna, pilih nama kebijakan yang ingin Anda lihat untuk memperluas ringkasan kebijakan.
3. Dalam ringkasan kebijakan, pilih nama layanan yang menyertakan tindakan yang tidak didukung.

4. Di sisi kiri di AWS Management Console footer, pilih Umpan Balik. Di IAM kotak Umpan Balik untuk, ketik**`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`**. Jika Anda ingin melaporkan lebih dari satu tindakan yang tidak didukung, ketik **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions`**.

Untuk meminta agar layanan lain mencakup tindakan yang hilang, ulangi tiga langkah terakhir.

Kebijakan saya tidak memberikan izin yang diharapkan

Untuk menetapkan izin bagi pengguna, grup, peran, atau sumber daya, Anda membuat kebijakan, yaitu dokumen yang mendefinisikan izin. Dokumen kebijakan mencakup elemen-elemen berikut ini:

- Efek – apakah kebijakan mengizinkan atau menolak akses
- Tindakan – daftar tindakan yang diperbolehkan atau ditolak oleh kebijakan tersebut
- Sumber Daya – daftar sumber daya yang memungkinkan terjadinya tindakan
- Kondisi (Opsional) – situasi yang mengatur pemberian izin oleh kebijakan

Untuk mempelajari tentang elemen kebijakan ini dan elemen kebijakan lainnya, lihat IAMJSONreferensi elemen kebijakan.

Untuk memberikan akses, kebijakan Anda harus menentukan tindakan dengan sumber daya yang didukung. Jika kebijakan Anda juga mencakup suatu kondisi, kondisi tersebut harus mencakup kunci kondisi global atau harus berlaku untuk tindakan tersebut. Untuk mempelajari sumber daya mana yang didukung oleh suatu tindakan, lihat AWS dokumentasi untuk layanan Anda. Untuk mempelajari kondisi mana yang didukung oleh tindakan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Periksa apakah kebijakan Anda menentukan tindakan, sumber daya, atau kondisi yang tidak memberikan izin. Lihat ringkasan kebijakan untuk kebijakan Anda menggunakan IAM konsol di https://console.aws.amazon.com/iam/. Anda dapat menggunakan ringkasan kebijakan untuk mengidentifikasi dan memperbaiki masalah di kebijakan Anda.

Ada beberapa alasan mengapa elemen mungkin tidak memberikan izin meskipun telah ditentukan dalam IAM kebijakan:

- Tindakan didefinisikan tanpa sumber daya yang berlaku

- Sumber daya didefinisikan tanpa tindakan yang berlaku
- Suatu kondisi didefinisikan tanpa tindakan yang berlaku

Untuk melihat contoh ringkasan kebijakan yang mencakup peringatan, lihat the section called "Ringkasan kebijakan (daftar layanan)".

Tindakan didefinisikan tanpa sumber daya yang sesuai

Kebijakan di bawah ini menjelaskan semua tindakan `ec2:Describe*` dan menentukan sumber daya spesifik. Tidak satu pun tindakan `ec2:Describe` diberikan karena semuanya tidak mendukung izin tingkat sumber daya. Izin tingkat sumber daya berarti bahwa tindakan tersebut mendukung sumber daya yang digunakan ARNsdalam elemen kebijakan. `Resource` Jika tindakan tidak mendukung izin tingkat sumber daya, maka pernyataan dalam kebijakan tersebut harus menggunakan wildcard (*) di elemen `Resource`. Untuk mempelajari layanan mana yang mendukung izin tingkat sumber daya, lihat AWS layanan yang bekerja dengan IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "arn:aws:ec2:us-east-2:ACCOUNT-ID:instance/*"
    }]
}
```

Kebijakan ini tidak memberikan izin apa pun, dan ringkasan kebijakan mencakup kesalahan berikut ini:

```
This policy does not grant any permissions. To grant access, policies must
have an action that has an applicable resource or condition.
```

Untuk memperbaiki kebijakan ini, Anda harus menggunakan * di elemen `Resource`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
```

```
}
```

Sumber daya ditetapkan tanpa tindakan yang sesuai

Kebijakan di bawah ini mendefinisikan sumber daya bucket Amazon S3 tetapi tidak mencakup tindakan S3 yang dapat dilakukan pada sumber daya tersebut. Kebijakan ini juga memberikan akses penuh ke semua CloudFront tindakan Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ]
    }]
}
```

Kebijakan ini memberikan izin untuk semua CloudFront tindakan. Tetapi karena kebijakan mendefinisikan sumber daya `amzn-s3-demo-bucket` S3 tanpa mendefinisikan tindakan S3 apa pun, ringkasan kebijakan mencakup peringatan berikut:

```
This policy defines some actions, resources, or conditions that do not
provide permissions. To grant access, policies must have an action that has
an applicable resource or condition.
```

Untuk memperbaiki kebijakan ini guna memberikan izin bucket S3, Anda harus menetapkan tindakan S3 yang dapat dilakukan pada sumber daya bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "cloudfront:*",
            "s3:CreateBucket",
            "s3:ListBucket*",
            "s3:PutBucket*",
            "s3:GetBucket*"
```

```
        ],
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ]
    }]
}
```

Sebagai alternatif, untuk memperbaiki kebijakan ini agar hanya memberikan CloudFront izin, hapus sumber daya S3.

Kondisi didefinisikan tanpa tindakan yang berlaku

Kebijakan di bawah ini mendefinisikan dua tindakan Amazon S3 untuk semua sumber daya S3, jika awalan S3 sama dan ID versi `custom` sama. `1234` Namun, kunci kondisi `s3:VersionId` digunakan untuk penandaan versi objek dan tidak didukung oleh tindakan bucket didefinisikan. Untuk mempelajari kondisi mana yang didukung oleh tindakan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan untuk melihat dokumentasi layanan untuk kunci kondisi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
```

```
}
```

Kebijakan ini memberikan izin untuk tindakan `s3:ListBucketVersions` dan tindakan `s3:ListBucket` jika nama bucket mencakup prefiks `custom`. Tetapi karena kondisi `s3:VersionId` tidak didukung oleh salah satu tindakan yang ditetapkan, ringkasan kebijakan mencakup kesalahan berikut:

```
This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.
```

Untuk memperbaiki kebijakan ini untuk menggunakan penandaan versi objek S3, Anda harus menentukan tindakan S3 yang mendukung kunci kondisi `s3:VersionId`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Kebijakan ini memberikan izin untuk setiap tindakan dan kondisi dalam kebijakan tersebut. Namun, kebijakan ini masih tidak memberikan izin apa pun karena tidak ada kasus ketika tindakan tunggal sesuai dengan kedua kondisi. Sebaliknya, Anda harus membuat dua pernyataan terpisah yang masing-masing hanya menyertakan tindakan kondisi yang sesuai dengannya.

Untuk memperbaiki kebijakan ini, buat dua pernyataan. Pernyataan pertama mencakup tindakan yang mendukung kondisi `s3:prefix`, dan pernyataan kedua mencakup tindakan yang mendukung kondisi `s3:VersionId`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

Pemecahan masalah manajemen kebijakan

Anda dapat mendiagnosis dan menyelesaikan masalah terkait manajemen kebijakan.

Melampirkan atau melepaskan kebijakan di akun IAM

Beberapa AWS kebijakan terkelola ditautkan ke layanan. Kebijakan ini hanya digunakan dengan peran yang berkaitan dengan layanan untuk layanan itu. Di IAM konsol, saat Anda melihat halaman Detail kebijakan, halaman tersebut menyertakan spanduk untuk menunjukkan bahwa kebijakan

tersebut ditautkan ke layanan. Anda tidak dapat melampirkan kebijakan ini ke pengguna, grup, atau peran di dalamnyaIAM. Saat Anda membuat peran terkait layanan, kebijakan ini secara otomatis dilampirkan ke peran baru Anda. Karena kebijakan tersebut diperlukan, Anda tidak dapat memisahkan kebijakan dari peran yang terkait dengan layanan.

Mengubah kebijakan IAM identitas Anda berdasarkan aktivitasnya

Anda dapat memperbarui kebijakan untuk IAM identitas Anda (pengguna, grup, dan peran) berdasarkan aktivitas mereka. Untuk melakukan ini, lihat peristiwa akun Anda di Riwayat CloudTrail acara. CloudTrail log peristiwa mencakup informasi peristiwa terperinci yang dapat Anda gunakan untuk mengubah izin kebijakan.

Seorang pengguna atau peran sedang mencoba untuk melakukan suatu tindakan di AWS dan permintaan itu ditolak.

- Pertimbangkan apakah pengguna atau peran harus memiliki izin untuk melakukan tindakan. Jika demikian, Anda dapat menambahkan tindakan dan bahkan sumber daya yang mereka coba akses ke kebijakan mereka. ARN

Pengguna atau peran memiliki izin yang tidak mereka gunakan.

- Pertimbangkan untuk menghapus izin tersebut dari kebijakan mereka. Pastikan bahwa kebijakan Anda memberikan hak istimewa terkecil yang diperlukan untuk hanya melakukan tindakan yang diperlukan.

Untuk informasi selengkapnya tentang penggunaan CloudTrail, lihat Melihat CloudTrail Acara di CloudTrail Konsol di AWS CloudTrail Panduan Pengguna.

Memecahkan masalah dokumen kebijakan JSON

Anda dapat mendiagnosis dan menyelesaikan masalah yang berkaitan dengan dokumen JSON kebijakan.

Memvalidasi kebijakan Anda

Saat Anda membuat atau mengedit JSON kebijakan, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAMmengidentifikasi kesalahan JSON sintaks, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari

selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat Validasi kebijakan IAMAccess Analyzer.

Saya tidak memiliki izin untuk validasi kebijakan di editor JSON

Dalam AWS Management Console, Anda mungkin menerima kesalahan berikut jika Anda tidak memiliki izin untuk melihat hasil validasi kebijakan IAM Access Analyzer:

`You need permissions. You do not have the permissions required to perform this operation. Ask your administrator to add permissions.`

Untuk memperbaiki kesalahan ini, minta administrator Anda untuk menambahkan izin `access-analyzer:ValidatePolicy` untuk Anda.

Lebih dari satu objek JSON kebijakan

IAMKebijakan harus terdiri dari hanya satu JSON objek. Anda menunjukkan sebuah objek dengan menempatkan rungkup { } di sekitarnya. Anda dapat membuat sarang objek lain di dalam JSON objek dengan menyematkan tanda kurung {} tambahan di dalam pasangan luar. Kebijakan harus berisi hanya satu pasang kurung kurung {} terluar. Contoh berikut ini salah karena berisi dua objek di tingkat atas (disebut dalam *red*):

```
{
      "Version": "2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
   }
   {
     "Statement": {
         "Effect": "Allow",
         "Action": "s3:*",
         "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
   }
```

Namun, Anda dapat memenuhi maksud dari contoh sebelumnya dengan menggunakan tata bahasa kebijakan yang benar. Alih-alih memasukkan dua objek kebijakan lengkap, masing-masing dengan elemen `Statement`, Anda dapat menggabungkan kedua blok menjadi satu elemen `Statement`.

Elemen `Statement` memiliki susunan dua objek sebagai nilainya, seperti ditunjukkan dalam contoh berikut (disebut dalam huruf tebal):

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":" *"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
      ]
    }
```

Lebih dari satu elemen JSON pernyataan

Kesalahan ini pada awalnya mungkin tampak seperti variasi pada bagian sebelumnya. Namun, secara sintaksis ini adalah tipe kesalahan yang berbeda. Contoh berikut hanya memiliki satu objek kebijakan yang ditandai dengan sepasang rungkup { } di tingkat atas. Namun, objek tersebut berisi dua elemen `Statement` di dalamnya.

IAMKebijakan harus berisi hanya satu `Statement` elemen, yang terdiri dari name (`Statement`) yang muncul di sebelah kiri titik dua, diikuti oleh nilainya di sebelah kanan. Nilai dari elemen `Statement` harus berupa objek, yang ditandai dengan rungkup { }, yang berisi satu elemen `Effect`, satu elemen `Action`, dan satu elemen `Resource`. Contoh berikut ini salah karena berisi dua elemen `Statement` dalam objek kebijakan (disebut dalam *red*):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
```

```
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Objek nilai dapat berupa himpunan berbagai objek nilai. Untuk memecahkan masalah ini, gabungkan kedua elemen `Statement` ke dalam satu elemen dengan himpunan objek, seperti ditunjukkan dalam contoh berikut (disebut dalam huruf tebal):

```
{
      "Version": "2012-10-17",
      "Statement": [    {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
     ]
    }
```

Nilai dari elemen `Statement` merupakan himpunan objek. Rangkaian dalam contoh ini terdiri dari dua objek, yang masing-masing adalah nilai yang benar untuk elemen `Statement`. Setiap objek di himpunan dipisahkan dengan koma.

Lebih dari satu efek, tindakan, atau elemen sumber daya dalam elemen JSON pernyataan

Di sisi nilai pasangan nama/nilai `Statement`, objek harus terdiri dari hanya satu elemen `Effect`, satu elemen `Action`, dan satu elemen `Resource`. Kebijakan berikut tidak benar karena memiliki dua `Effect` elemen dalam`Statement`:

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
```

```
      "Resource": "*"
    }
  }
```

> **Note**
>
> Mesin kebijakan tidak mengizinkan kesalahan tersebut di kebijakan baru atau yang telah diedit. Namun demikian, mesin kebijakan terus mengizinkan kebijakan yang disimpan sebelum mesin diperbarui. Perilaku kebijakan yang ada dengan kesalahan tersebut adalah sebagai berikut:
>
> - Beberapa elemen `Effect`: hanya elemen `Effect` terakhir yang diamati. Yang lainnya diabaikan.
> - Beberapa `Action` elemen: semua `Action` elemen digabungkan secara internal dan diperlakukan seolah-olah mereka adalah daftar tunggal.
> - Beberapa `Resource` elemen: semua `Resource` elemen digabungkan secara internal dan diperlakukan seolah-olah mereka adalah daftar tunggal.
>
> Mesin kebijakan tidak memungkinkan Anda untuk menyimpan kebijakan dengan kesalahan sintaksis. Perbaiki kesalahan dalam kebijakan sebelum menyimpan. Tinjau dan perbaiki rekomendasi validasi kebijakan apa pun untuk kebijakan Anda.

Dalam setiap kasus, solusinya adalah menghapus elemen ekstra yang salah. Untuk `Effect` elemen, ini mudah: jika Anda ingin contoh sebelumnya menolak izin ke EC2 instans Amazon, maka Anda harus menghapus baris `"Effect": "Allow",` dari kebijakan, sebagai berikut:

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

Namun, jika elemen duplikat adalah `Action` atau `Resource`, maka resolusinya dapat lebih rumit. Anda mungkin memiliki beberapa tindakan yang ingin Anda izinkan (atau tolak) izinnya, atau Anda

mungkin ingin mengontrol akses ke beberapa sumber daya. Misalnya, contoh berikut salah karena memiliki beberapa elemen `Resource` (yang disebut dalam *red*):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Masing-masing elemen wajib di objek nilai elemen `Statement` hanya bisa muncul sekali. Solusinya adalah menempatkan setiap nilai dalam himpunan. Contoh berikut menggambarkan hal ini dengan membuat dua elemen sumber daya terpisah menjadi satu elemen `Resource` dengan susunan sebagai objek nilai (disebut dalam huruf tebal):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket",
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
      }
    }
```

Elemen JSON versi yang hilang

Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Sebagai perbandingan, versi kebijakan dibuat saat Anda mengubah kebijakan yang dikelola pelangganIAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola. Untuk mempelajari selengkapnya tentang elemen kebijakan `Version`, lihat IAMJSONelemen kebijakan: Version. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called “Kebijakan IAM versioning”.

Sebagai AWS fitur berkembang, kemampuan baru ditambahkan ke IAM kebijakan untuk mendukung fitur tersebut. Terkadang, pembaruan sintaksis kebijakan mencakup nomor versi baru. Jika Anda menggunakan fitur tata bahasa kebijakan yang lebih baru dalam kebijakan Anda, maka Anda harus memberi tahu mesin pembatas kebijakan versi mana yang Anda gunakan. Versi kebijakan default adalah “17-10-2008.” Jika Anda ingin menggunakan fitur kebijakan yang diperkenalkan setelahnya, Anda harus menentukan nomor versi yang mendukung fitur yang Anda inginkan. Sebaiknya Anda selalu menyertakan nomor versi sintaksis kebijakan terbaru, yang saat ini adalah `"Version": "2012-10-17"`. Misalnya, kebijakan berikut tidak benar karena menggunakan variabel kebijakan `${...}` di ARN sumber daya. Tetapi kebijakan ini gagal menentukan versi sintaksis kebijakan yang mendukung variabel kebijakan (disebut dalam *red*):

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Menambahkan `Version` elemen di bagian atas kebijakan dengan nilai`2012-10-17`, IAM API versi pertama yang mendukung variabel kebijakan, memecahkan masalah ini (dipanggil dalam huruf tebal):

```
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Memecahkan masalah kunci keamanan FIDO

Gunakan informasi di sini untuk membantu Anda mendiagnosis masalah umum yang mungkin Anda temui saat bekerja dengan kunci FIDO2 keamanan.

Topik

Saya tidak dapat mengaktifkan kunci FIDO keamanan saya

Konsultasikan solusi berikut tergantung pada status Anda sebagai IAM pengguna atau administrator sistem

IAMpengguna

Jika Anda tidak dapat mengaktifkan kunci FIDO keamanan, periksa hal berikut:

- Apakah Anda menggunakan konfigurasi yang didukung?

  Untuk informasi tentang perangkat dan browser yang dapat Anda gunakan WebAuthn dan AWS, lihatKonfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan.

- Apakah Anda menggunakan Mozilla Firefox?

  Versi Firefox saat ini mendukung secara WebAuthn default. Untuk mengaktifkan dukungan WebAuthn di Firefox, lakukan hal berikut:

  1. Dari bilah alamat Firefox, ketik **`about:config`**.
  2. Pada bilah Pencarian dari layar yang terbuka, ketik **`webauthn`**.
  3. Pilih security.webauth.webauthn dan ubah nilainya menjadi true.

- Apakah Anda menggunakan plugin peramban?

  AWS tidak mendukung penggunaan plugin untuk menambahkan dukungan WebAuthn browser. Sebagai gantinya, gunakan browser yang menawarkan dukungan asli WebAuthn standar.

  Bahkan jika Anda menggunakan browser yang didukung, Anda mungkin memiliki plugin yang tidak kompatibel dengannya WebAuthn. Plugin yang tidak kompatibel dapat mencegah Anda mengaktifkan dan menggunakan kunci keamanan FIDO yang sesuai. Nonaktifkan plugin apa pun yang mungkin tidak kompatibel dan restart browser Anda. Kemudian, coba lagi aktifkan kunci FIDO keamanan.

- Apakah Anda memiliki izin yang sesuai?

Jika tidak ada satu pun masalah kompatibilitas di atas, Anda mungkin tidak memiliki izin yang sesuai. Hubungi administrator sistem Anda.

Administrator sistem

Jika IAM pengguna Anda tidak dapat mengaktifkan kunci FIDO keamanannya meskipun menggunakan konfigurasi yang didukung, periksa izin mereka. Untuk contoh terperinci, lihat IAMtutorial: Izinkan pengguna untuk mengelola kredensi dan pengaturan mereka MFA.

Saya tidak bisa masuk menggunakan kunci FIDO keamanan

Jika Anda tidak dapat masuk AWS Management Console menggunakan kunci FIDO keamanan, lihat duluKonfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan. Jika Anda menggunakan konfigurasi yang didukung tetapi tidak dapat masuk, hubungi administrator sistem untuk mendapatkan bantuan.

Saya kehilangan atau merusak kunci FIDO keamanan saya

Hingga delapan MFA perangkat dari kombinasi MFAjenis yang saat ini didukung dapat ditetapkan ke pengguna. Dengan beberapa MFA perangkat, Anda hanya perlu satu MFA perangkat untuk masuk ke perangkat AWS Management Console. Mengganti kunci FIDO keamanan mirip dengan mengganti TOTP token perangkat keras. Jika Anda kehilangan atau merusak semua jenis MFA perangkat, lihatMemulihkan identitas yang MFA dilindungi di IAM.

Masalah lainnya

Jika Anda memiliki masalah dengan kunci FIDO keamanan yang tidak tercakup di sini, lakukan salah satu hal berikut:

- Pengguna IAM: Hubungi administrator sistem Anda.
- Akun AWS pengguna root: Hubungi AWS Support.

Memecahkan masalah peran IAM

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan IAM peran.

Topik

Saya tidak dapat mengsumsikan peran

Periksa hal berikut:

- Untuk memungkinkan pengguna untuk mengambil peran saat ini lagi dalam sesi peran, tentukan peran ARN atau Akun AWS ARN sebagai prinsipal dalam kebijakan kepercayaan peran. Layanan AWS yang menyediakan sumber daya komputasi seperti AmazonEC2, Amazon, Amazon ECSEKS, dan Lambda memberikan kredensi sementara dan secara otomatis memperbarui kredensi ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensional yang valid. Untuk layanan ini, tidak perlu mengambil peran saat ini lagi untuk mendapatkan kredensi sementara. Namun, jika Anda berniat untuk meneruskan tag sesi atau kebijakan sesi, Anda perlu mengambil peran saat ini lagi. Untuk mempelajari cara memodifikasi kebijakan kepercayaan peran untuk menambahkan peran utama ARN atau Akun AWS ARN, lihatMemperbarui kebijakan kepercayaan peran .
- Ketika Anda mengambil peran menggunakan AWS Management Console, pastikan untuk menggunakan nama yang tepat dari peran Anda. Nama peran peka huruf besar/kecil.
- Ketika Anda mengambil peran menggunakan AWS STS API atau AWS CLI, pastikan untuk menggunakan nama yang tepat dari peran Anda diARN. Nama peran peka huruf besar/kecil.
- Verifikasi bahwa IAM kebijakan Anda memberi Anda izin `sts:AssumeRole` untuk memanggil peran yang ingin Anda ambil. `Action`Elemen IAM kebijakan Anda harus memungkinkan Anda untuk memanggil `AssumeRole` tindakan. Selain itu, `Resource` elemen IAM kebijakan Anda harus menentukan peran yang ingin Anda ambil. Misalnya, `Resource` elemen dapat menentukan peran

berdasarkan Amazon Resource Name (ARN) atau dengan wildcard (*). Misalnya, setidaknya satu kebijakan yang berlaku bagi Anda harus memberikan izin yang serupa dengan yang berikut ini:

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Verifikasi bahwa IAM identitas Anda ditandai dengan tag apa pun yang diperlukan oleh IAM kebijakan tersebut. Misalnya, dalam izin kebijakan berikut, elemen `Condition` mewajibkan Anda, sebagai prinsipal yang meminta untuk mengasumsikan peran tersebut, memiliki tanda khusus. Anda harus ditandai dengan `department = HR` atau `department = CS`. Jika tidak, Anda tidak dapat mengambil peran tersebut. Untuk mempelajari tentang menandai IAM pengguna dan peran, lihatthe section called “Tag untuk IAM sumber daya”.

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "*",
"Condition": {"StringEquals": {"aws:PrincipalTag/department": [
        "HR",
        "CS"
    ]}}
```

- Verifikasi bahwa Anda memenuhi semua kondisi yang ditentukan di kebijakan kepercayaan peran tersebut. `Condition` dapat menentukan tanggal kedaluwarsa, ID eksternal, atau bahwa permintaan harus berasal hanya dari alamat IP tertentu. Pertimbangkan contoh berikut: Jika tanggal saat ini adalah kapan saja setelah tanggal yang ditentukan, kebijakan tidak pernah cocok dan tidak dapat memberi Anda izin untuk mengambil peran tersebut.

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
"Condition": {
    "DateLessThan" : {
        "aws:CurrentTime" : "2016-05-01T12:00:00Z"
    }
}
```

- Verifikasi bahwa Akun AWS dari mana Anda menelepon `AssumeRole` adalah entitas tepercaya untuk peran yang Anda asumsikan. Entitas tepercaya didefinisikan sebagai `Principal` di kebijakan kepercayaan peran. Contoh berikut adalah kebijakan kepercayaan yang terlampir pada

peran yang ingin Anda asumsikan. Dalam contoh ini, ID akun dengan IAM pengguna yang Anda masuki harus 123456789012. Jika nomor akun Anda tidak tercantum di elemen `Principal` kebijakan kepercayaan peran, maka Anda tidak dapat mengasumsikan peran tersebut. Izin apa pun yang diberikan kepada Anda dalam kebijakan akses tidak akan berpengaruh. Perhatikan bahwa kebijakan contoh membatasi izin untuk tindakan yang terjadi antara 1 Juli 2017 dan 31 Desember 2017 (UTC), inklusif. Jika Anda masuk sebelum atau sesudah tanggal-tanggal tersebut, maka kebijakan tidak cocok, dan Anda tidak dapat mengasumsikan peran tersebut.

```
"Effect": "Allow",
"Principal": { "AWS": "arn:aws:iam::123456789012:root" },
"Action": "sts:AssumeRole",
"Condition": {
  "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
  "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
}
```

- Identitas Sumber - Administrator dapat mengonfigurasi peran untuk meminta identitas meneruskan string kustom yang mengidentifikasi orang atau aplikasi yang melakukan tindakan AWS, yang disebut identitas sumber. Verifikasi apakah peran yang diambil mengharuskan identitas sumber ditetapkan. Untuk informasi selengkapnya tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Peran baru muncul di akun AWS saya

Beberapa AWS layanan mengharuskan Anda menggunakan jenis peran layanan unik yang ditautkan langsung ke layanan. Peran yang berkaitan dengan layanan ini telah ditetapkan sebelumnya oleh layanan dan mencakup semua izin yang diperlukan layanan. Hal ini memudahkan pengaturan layanan karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Untuk informasi umum tentang peran terkait layanan, lihat Buat peran tertaut layanan.

Anda mungkin sudah menggunakan layanan ketika layanan mulai mendukung peran yang terkait dengan layanan. Jika demikian, Anda mungkin menerima surel tentang peran baru di akun Anda. Peran ini mencakup semua izin yang diperlukan layanan untuk melakukan tindakan atas nama Anda. Anda tidak perlu mengambil tindakan apa pun untuk mendukung peran ini. Namun, Anda tidak boleh menghapus peran tersebut dari akun Anda. Melakukan hal tersebut dapat menghapus izin yang diperlukan layanan untuk mengakses sumber daya AWS . Anda dapat melihat peran yang ditautkan layanan di akun Anda dengan membuka halaman IAM Peran konsol. IAM Peran yang terkait dengan layanan muncul dengan (Peran yang terkait dengan layanan) di kolom Entitas tepercaya di tabel.

Untuk informasi tentang layanan mana yang mendukung peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran yang Terkait dengan Layanan. Untuk informasi tentang menggunakan peran yang terkait dengan layanan, pilih tautan Ya.

Saya tidak dapat mengedit atau menghapus peran di Akun AWS

Anda tidak dapat menghapus atau mengedit izin untuk peran terkait layanan di. IAM Peran ini mencakup kepercayaan dan izin yang ditetapkan sebelumnya yang diperlukan oleh layanan untuk melakukan tindakan atas nama Anda. Anda dapat menggunakan IAM konsol AWS CLI,, atau API hanya mengedit deskripsi peran terkait layanan. Anda dapat melihat peran terkait layanan di akun Anda dengan membuka halaman IAM Peran di konsol. Peran yang terkait dengan layanan muncul dengan (Peran yang terkait dengan layanan) di kolom Entitas tepercaya di tabel. Banner pada halaman Ringkasan peran juga menunjukkan bahwa peran tersebut adalah peran yang terkait dengan layanan. Anda dapat mengelola dan menghapus peran ini hanya melalui layanan yang terhubung, jika layanan tersebut mendukung tindakan. Berhati-hatilah saat memodifikasi atau menghapus peran terkait layanan karena melakukan hal ini dapat menghapus izin yang diperlukan layanan untuk mengakses sumber daya AWS .

Untuk informasi tentang layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran yang Terkait dengan Layanan.

Saya tidak berwenang untuk melakukan: iam: PassRole

Saat Anda membuat peran yang terkait dengan layanan, Anda harus memiliki izin untuk meneruskan peran tersebut ke layanan. Beberapa layanan secara otomatis membuat peran yang terkait dengan layanan di akun Anda ketika Anda melakukan tindakan di layanan tersebut. Misalnya, Amazon EC2 Auto Scaling membuat peran `AWSServiceRoleForAutoScaling` terkait layanan untuk Anda saat pertama kali membuat grup Auto Scaling. Jika Anda mencoba membuat grup Auto Scaling tanpa izin `PassRole`, Anda menerima kesalahan berikut ini:

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

Untuk memperbaiki kesalahan ini, minta administrator Anda untuk menambahkan izin `iam:PassRole` untuk Anda.

Untuk mempelajari layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari apakah layanan secara otomatis membuat peran yang terkait dengan layanan untuk Anda, pilih tautan Ya untuk melihat dokumentasi peran layanan yang terkait dengan layanan.

Mengapa saya tidak dapat mengasumsikan peran dengan sesi 12 jam? (AWS CLI, AWS API)

Saat Anda menggunakan `assume-role*` CLI operasi AWS STS `AssumeRole*` API atau untuk mengambil peran, Anda dapat menentukan nilai untuk `DurationSeconds` parameter tersebut. Anda dapat menentukan nilai dari 900 detik (15 menit) hingga pengaturan Durasi sesi maksimum untuk peran tersebut. Jika Anda menentukan nilai yang lebih tinggi dari pengaturan ini, operasi akan gagal. Pengaturan ini dapat memiliki nilai maksimum 12 jam. Misalnya, jika Anda menentukan durasi sesi 12 jam, tetapi administrator Anda mengatur durasi sesi maksimum menjadi 6 jam, operasi Anda gagal. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Jika Anda menggunakan rantai peran (menggunakan peran untuk mengsumsikan peran kedua), sesi Anda dibatasi hingga maksimum satu jam. Jika Anda kemudian menggunakan parameter `DurationSeconds` untuk memberikan nilai lebih dari satu jam, operasi akan gagal.

Saya menerima kesalahan ketika saya mencoba beralih peran di IAM konsol

Informasi yang Anda masukkan di halaman Beralih Peran harus sesuai dengan informasi untuk peran. Jika tidak, operasi akan gagal dan Anda menerima kesalahan berikut:

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

Jika Anda menerima pesan kesalahan ini, pastikan bahwa informasi berikut benar:

- ID Akun atau alias — Akun AWS ID adalah nomor 12 digit. Akun Anda mungkin memiliki alias, yang merupakan pengenal ramah seperti nama perusahaan Anda yang dapat digunakan sebagai pengganti ID Anda Akun AWS . Anda dapat menggunakan ID akun atau alias di bidang ini.
- Nama peran – Nama peran bersifat peka terhadap kapitalisasi huruf. ID akun dan nama peran harus sesuai dengan apa yang dikonfigurasi untuk peran tersebut.

Jika Anda terus menerima pesan kesalahan, hubungi administrator Anda untuk memverifikasi informasi sebelumnya. Kebijakan kepercayaan peran atau kebijakan IAM pengguna mungkin membatasi akses Anda. Administrator Anda dapat memverifikasi izin untuk kebijakan ini.

Peran saya memiliki kebijakan yang memungkinkan saya melakukan tindakan, tetapi saya mendapatkan “akses ditolak”

Sesi peran Anda mungkin dibatasi oleh kebijakan sesi. Bila Anda meminta kredenial keamanan sementara secara terprogram menggunakan AWS STS, Anda dapat secara opsional meneruskan kebijakan sesi inline atau terkelola. Kebijakan sesi adalah kebijakan lanjutan yang Anda sampaikan sebagai parameter saat Anda secara terprogram membuat sesi kredensial sementara untuk peran. Anda dapat meneruskan satu dokumen kebijakan sesi JSON inline menggunakan `Policy` parameter. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas dan kebijakan sesi peran. Selain itu, jika administrator Anda atau program khusus memberi Anda kredensial sementara, mereka mungkin telah menyertakan kebijakan sesi untuk membatasi akses Anda.

Layanan tidak membuat versi kebijakan default peran tersebut

Peran layanan adalah peran yang diasumsikan oleh layanan untuk melakukan tindakan di akun Anda atas nama Anda. Ketika Anda mengatur beberapa lingkungan AWS layanan, Anda harus menentukan peran untuk layanan untuk mengambil alih. Dalam beberapa kasus, layanan menciptakan peran layanan dan kebijakannya IAM untuk Anda. Meskipun Anda dapat memodifikasi atau menghapus peran layanan dan kebijakannya dari dalamIAM, AWS tidak merekomendasikan hal ini. Peran dan kebijakan ditujukan hanya untuk digunakan oleh layanan tersebut. Jika Anda mengedit kebijakan dan menyiapkan lingkungan lain, ketika layanan mencoba menggunakan peran dan kebijakan yang sama, operasi dapat gagal.

Misalnya, saat Anda menggunakan AWS CodeBuild untuk pertama kalinya, layanan akan membuat peran bernama`codebuild-RWBCore-service-role`. Peran layanan tersebut menggunakan kebijakan bernama `codebuild-RWBCore-managed-policy`. Jika Anda mengedit kebijakan, ini akan membuat versi baru dan menyimpan versi tersebut sebagai versi default. Jika Anda melakukan operasi berikutnya di AWS CodeBuild, layanan mungkin mencoba memperbarui kebijakan. Jika iya, Anda akan menerima kesalahan berikut:

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
```

`RWBCore-service-role role. To continue, detach the policy from any other identities and then delete the policy and the role.`

Jika Anda menerima kesalahan ini, Anda harus membuat perubahan IAM sebelum Anda dapat melanjutkan operasi layanan Anda. Pertama, atur versi kebijakan default ke V1 dan coba kembali operasi tersebut. Jika V1 sebelumnya dihapus, atau jika memilih V1 tidak berfungsi, maka bersihkan dan hapus kebijakan dan peran yang ada.

Untuk informasi selengkapnya tentang mengedit kebijakan terkelola, lihat Menyunting kebijakan terkelola pelanggan (konsol). Untuk informasi selengkapnya tentang versi kebijakan, lihat Kebijakan IAM versioning.

Untuk menghapus peran layanan dan kebijakannya

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda hapus.
4. Pilih tab Entitas yang dilampirkan untuk melihat IAM pengguna, grup, atau peran mana yang menggunakan kebijakan ini. Jika salah satu dari identitas ini menggunakan kebijakan, selesaikan tugas berikut:
    a. Buat kebijakan terkelola baru dengan izin yang diperlukan. Untuk memastikan bahwa identitas memiliki izin yang sama sebelum dan sesudah tindakan Anda, salin dokumen JSON kebijakan dari kebijakan yang ada. Kemudian buat kebijakan terkelola baru dan tempel JSON dokumen seperti yang dijelaskan dalam Membuat Kebijakan menggunakan JSON editor.
    b. Untuk setiap identitas yang terpengaruh, lampirkan kebijakan baru dan lepaskan kebijakan lama. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.
5. Di panel navigasi, pilih Peran.
6. Dalam daftar peran, pilih nama peran yang ingin Anda hapus.
7. Pilih tab Hubungan kepercayaan untuk melihat entitas mana yang dapat mengasumsikan peran tersebut. Jika entitas selain layanan dicantumkan, selesaikan tugas berikut:
    a. Buat peran baru yang memercayai entitas tersebut.
    b. Kebijakan yang Anda buat pada langkah sebelumnya. Jika Anda melewatkan langkah tersebut, buat kebijakan terkelola baru sekarang.

c. Beri tahu siapa pun yang mengasumsikan peran tersebut bahwa mereka tidak dapat lagi melakukannya. Berikan informasi kepada mereka tentang cara mengasumsikan peran baru dan memiliki izin yang sama.

8. Hapus kebijakan.
9. Hapus peran.

Tidak ada kasus penggunaan untuk peran layanan di konsol

Beberapa layanan mengharuskan Anda membuat peran layanan secara manual untuk memberi layanan izin untuk melakukan tindakan atas nama Anda. Jika layanan tidak terdaftar di IAM konsol, Anda harus mencantumkan layanan secara manual sebagai prinsipal tepercaya. Jika dokumentasi untuk layanan atau fitur yang Anda gunakan tidak menyertakan petunjuk untuk mencantumkan layanan sebagai prinsipal tepercaya, berikan umpan balik untuk halaman.

Untuk membuat peran layanan secara manual, Anda harus mengetahui prinsipal layanan untuk layanan yang akan mengambil peran. Sebuah prinsipal layanan adalah pengidentifikasi yang digunakan untuk memberikan izin layanan. Prinsipal layanan ditentukan oleh layanan tersebut.

Anda dapat menemukan prinsipal layanan untuk beberapa layanan dengan memeriksa hal berikut:

1. Buka AWS layanan yang bekerja dengan IAM.
2. Periksa apakah layanan memiliki Yes (Ya) di kolom Service-linked roles (Peran terkait layanan).
3. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.
4. Temukan bagian Izin peran terkait layanan untuk layanan tersebut untuk melihat prinsipal layanan.

Anda dapat membuat peran layanan secara manual menggunakan AWS CLI perintah atau AWS APIoperasi. Untuk membuat peran layanan secara manual menggunakan IAM konsol, selesaikan tugas-tugas berikut:

1. Buat IAM peran menggunakan ID akun Anda. Jangan lampirkan kebijakan atau beri izin apa pun. Untuk detailnya, lihat Membuat peran untuk mendelegasikan izin ke pengguna IAM.
2. Buka peran dan edit hubungan kepercayaan. Alih-alih memercayai akun, peran harus memercayai layanan. Misalnya, perbarui yang elemen `Principal` berikut:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Ubah prinsipal menjadi nilai untuk layanan Anda, sepertiIAM.

```
"Principal": { "Service": "iam.amazonaws.com" }
```

3. Tambahkan izin yang diperlukan layanan dengan melampirkan kebijakan izin untuk peran.
4. Kembali ke layanan yang memerlukan izin dan gunakan metode yang didokumentasikan untuk memberi tahu layanan tentang peran layanan baru.

Memecahkan masalah IAM dan Amazon EC2

Informasi berikut dapat membantu Anda memecahkan IAM masalah dengan Amazon. EC2

Topik

- Saat saya mencoba meluncurkan instance, saya tidak melihat peran tersebut di daftar IAMPeran EC2 konsol Amazon
- Kredensial pada instans saya untuk peran yang salah
- Saat saya mencoba memanggil AddRoleToInstanceProfile, saya mendapatkan kesalahan AccessDenied
- AmazonEC2: Ketika saya mencoba meluncurkan instance dengan peran, saya mendapatkan AccessDenied kesalahan
- Saya tidak dapat mengakses kredensi keamanan sementara pada instance saya EC2
- Apa arti kesalahan dari info dokumen di IAM subpohon?

Saat saya mencoba meluncurkan instance, saya tidak melihat peran tersebut di daftar IAMPeran EC2 konsol Amazon

Periksa hal-hal berikut:

- Jika Anda masuk sebagai IAM pengguna, verifikasi bahwa Anda memiliki izin untuk meneleponListInstanceProfiles. Untuk informasi tentang izin yang diperlukan untuk bekerja dengan peran, lihatIzin diperlukan untuk menggunakan peran dengan Amazon EC2. Untuk informasi tentang menambahkan izin ke pengguna, lihat Kelola IAM kebijakan.

  Jika Anda tidak dapat mengubah izin Anda sendiri, Anda harus menghubungi administrator yang dapat bekerja dengan IAM untuk memperbarui izin Anda.

- Jika Anda membuat peran menggunakan IAM CLI atauAPI, verifikasi hal berikut:
  - Anda membuat profil instans dan menambahkan peran ke profil instance tersebut.
  - Anda menggunakan nama yang sama untuk peran dan profil instance. Jika Anda memberi nama profil peran dan instans secara berbeda, Anda tidak akan melihat nama peran yang benar di EC2 konsol Amazon.

  Daftar IAMPeran di EC2 konsol Amazon mencantumkan nama profil instans, bukan nama peran. Anda harus memilih nama profil instans yang berisi peran yang Anda inginkan. Untuk detail tentang profil instans, lihat Gunakan profil contoh.

  > **Note**
  >
  > Jika Anda menggunakan IAM konsol untuk membuat peran, Anda tidak perlu bekerja dengan profil instance. Untuk setiap peran yang Anda buat di IAM konsol, profil instance dibuat dengan nama yang sama dengan peran, dan peran tersebut secara otomatis ditambahkan ke profil instance tersebut. Profil instance hanya dapat berisi satu IAM peran, dan batas itu tidak dapat ditingkatkan.

Kredensial pada instans saya untuk peran yang salah

Peran dalam profil instans mungkin telah diganti baru-baru ini. Jika demikian, aplikasi Anda perlu menunggu rotasi kredensial terjadwal otomatis berikutnya sebelum kredensial untuk peran Anda tersedia.

Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Saat saya mencoba memanggil **`AddRoleToInstanceProfile`**, saya mendapatkan kesalahan **`AccessDenied`**

Jika Anda membuat permintaan sebagai IAM pengguna, verifikasi bahwa Anda memiliki izin berikut:

- `iam:AddRoleToInstanceProfile`dengan sumber daya yang cocok dengan profil instance ARN (misalnya,`arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`).

Untuk informasi selengkapnya tentang izin yang diperlukan untuk bekerja dengan peran, lihatBagaimana saya memulainya?. Untuk informasi tentang menambahkan izin ke pengguna, lihat Kelola IAM kebijakan.

AmazonEC2: Ketika saya mencoba meluncurkan instance dengan peran, saya mendapatkan **`AccessDenied`** kesalahan

Periksa hal-hal berikut:

- Luncurkan instans tanpa profil instans. Ini akan membantu memastikan bahwa masalahnya terbatas pada IAM peran untuk EC2 instans Amazon.
- Jika Anda membuat permintaan sebagai IAM pengguna, verifikasi bahwa Anda memiliki izin berikut:
  - `ec2:RunInstances` dengan sumber daya wildcard ("*")
  - `iam:PassRole`dengan sumber daya yang cocok dengan peran ARN (misalnya,`arn:aws:iam::999999999999:role/ExampleRoleName`)
- Panggil IAM `GetInstanceProfile` tindakan untuk memastikan bahwa Anda menggunakan nama profil instans yang valid atau profil instans yang validARN. Untuk informasi selengkapnya, lihat Menggunakan IAM peran dengan EC2 instans Amazon.
- Panggil IAM `GetInstanceProfile` tindakan untuk memastikan bahwa profil instance memiliki peran. Profil instans kosong akan gagal dengan kesalahan `AccessDenied`. Untuk informasi selengkapnya tentang membuat peran, lihat IAMpenciptaan peran.

Untuk informasi selengkapnya tentang izin yang diperlukan untuk bekerja dengan peran, lihatBagaimana saya memulainya?. Untuk informasi tentang menambahkan izin ke pengguna, lihat Kelola IAM kebijakan.

Saya tidak dapat mengakses kredensi keamanan sementara pada instance saya EC2

Untuk mengakses kredenal keamanan sementara pada EC2 instans Anda, Anda harus terlebih dahulu menggunakan IAM konsol untuk membuat peran. Kemudian Anda meluncurkan EC2 instance yang menggunakan peran itu dan memeriksa instance yang sedang berjalan. Untuk informasi selengkapnya, lihat Bagaimana Cara Memulai? di Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2.

Jika Anda masih tidak dapat mengakses kredensi keamanan sementara pada EC2 instans Anda, periksa hal berikut:

- Bisakah Anda mengakses bagian lain dari Layanan Metadata Instans ()? IMDS Jika tidak, periksa apakah Anda tidak memiliki aturan firewall yang memblokir akses ke permintaan keIMDS.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/
hostname; echo
```

- Apakah `iam` subpohon dari yang IMDS ada? Jika tidak, verifikasi bahwa instans Anda memiliki profil IAM instance yang terkait dengannya dengan memanggil EC2 `DescribeInstances` API operasi atau menggunakan `aws ec2 describe-instances` CLI perintah.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- Periksa `info` dokumen di IAM subpohon untuk kesalahan. Jika Anda memiliki kesalahan, lihat Apa arti kesalahan dari `info` dokumen di IAM subpohon? untuk informasi selengkapnya.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

Apa arti kesalahan dari **`info`** dokumen di IAM subpohon?

Dokumen **`iam/info`** berarti **`"Code":"InstanceProfileNotFound"`**

Profil IAM instans Anda telah dihapus dan Amazon tidak EC2 dapat lagi memberikan kredensi ke instans Anda. Anda harus melampirkan profil instans yang valid ke EC2 instans Amazon Anda.

Jika ada profil instans dengan nama tersebut, periksa apakah profil instans tidak dihapus dan penggantinya dibuat dengan nama yang sama:

1. Panggil IAM `GetInstanceProfile` operasi untuk mendapatkan`InstanceProfileId`.
2. Hubungi EC2 `DescribeInstances` operasi Amazon untuk mendapatkan contohnya. `IamInstanceProfileId`
3. Verifikasi bahwa `InstanceProfileId` dari IAM operasi cocok dengan `IamInstanceProfileId` dari EC2 operasi Amazon.

Jika IDs berbeda, maka profil instance yang dilampirkan ke instans Anda tidak lagi valid. Anda harus melampirkan profil instans yang valid ke instans.

Dokumen **`iam/info`** menunjukkan keberhasilan tetapi juga menunjukkan **`"Message":"Instance Profile does not contain a role..."`**

Peran telah dihapus dari profil instance oleh IAM `RemoveRoleFromInstanceProfile` tindakan. Anda dapat menggunakan IAM `AddRoleToInstanceProfile` tindakan untuk melampirkan peran ke profil instance. Aplikasi Anda perlu menunggu hingga penyegaran yang terjadwal berikutnya untuk mengakses kredensial untuk peran tersebut.

Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Dokumen **`iam/security-credentials/[role-name]`** berarti **`"Code":"AssumeRoleUnauthorizedAccess"`**

Amazon EC2 tidak memiliki izin untuk mengambil peran tersebut. Izin untuk mengambil peran tersebut dikendalikan oleh kebijakan kepercayaan yang terlampir pada peran tersebut, seperti contoh berikut. Gunakan IAM `UpdateAssumeRolePolicy` API untuk memperbarui kebijakan kepercayaan.

```
{"Version": "2012-10-17","Statement": [{"Effect": "Allow","Principal": {"Service":
 ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

Aplikasi Anda perlu menunggu hingga penyegaran terjadwal otomatis berikutnya untuk mengakses kredensial untuk peran tersebut.

Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Memecahkan masalah IAM dan Amazon S3

Gunakan informasi di sini untuk membantu Anda memecahkan masalah dan memperbaiki masalah yang mungkin Anda temui saat bekerja dengan Amazon S3 dan. IAM

Bagaimana saya memberikan akses anonim ke bucket Amazon S3?

Anda menggunakan kebijakan bucket Amazon S3 yang menentukan wildcard (*) di elemen `principal`, yang berarti siapa pun dapat mengakses bucket. Dengan akses anonim, siapa pun

(termasuk pengguna tanpa Akun AWS) akan dapat mengakses bucket. Untuk kebijakan contoh, lihat Contoh Kasus untuk Kebijakan Bucket Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Saya masuk sebagai pengguna Akun AWS root. Mengapa saya tidak dapat mengakses bucket Amazon S3 di bawah akun saya?

Dalam beberapa kasus, Anda mungkin memiliki IAM pengguna dengan akses penuh ke IAM dan Amazon S3. Jika IAM pengguna menetapkan kebijakan bucket ke bucket Amazon S3 dan tidak menetapkan pengguna root sebagai prinsipal, pengguna root akan ditolak akses ke bucket tersebut. Namun, sebagai pengguna utama, Anda masih dapat mengakses bucket. Untuk melakukannya, ubah kebijakan bucket untuk mengizinkan akses pengguna root dari konsol Amazon S3 atau konsol. AWS CLI Gunakan penanggung jawab berikut, menggantikan *123456789012* dengan ID dari Akun AWS.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Memecahkan masalah SAML federasi dengan IAM

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah yang mungkin Anda temui saat bekerja dengan SAML 2.0 dan federasi AWS Identity and Access Management.

Topik

- Kesalahan: Permintaan Anda menyertakan respons yang tidak validSAML. Untuk logout, klik di sini.
- Kesalahan: RoleSessionName diperlukan dalam AuthnResponse (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan: InvalidIdentityToken)
- Kesalahan: Tidak berwenang untuk melakukan sts: AssumeRoleWith SAML (layanan: AWSSecurityTokenService; kode status: 403; kode kesalahan: AccessDenied)
- Kesalahan: RoleSessionName di AuthnResponse harus cocok [A-za-Z_0-9+=, .@-] {2,64} (layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService) InvalidIdentityToken
- Kesalahan: Identitas Sumber harus cocok dengan [A-za-Z_0-9+=, .@-] {2,64} dan tidak dimulai dengan "aws:" (layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService) InvalidIdentityToken
- Kesalahan: Tanda tangan respons tidak valid (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan:) InvalidIdentityToken

- Kesalahan: Gagal mengambil peran: Penerbit tidak hadir di penyedia tertentu (layanan: AWSOpenIdDiscoveryService; kode status: 400; kode kesalahan: AuthSamlInvalidSamlResponseException)
- Kesalahan: Tidak dapat mengurai metadata.
- Kesalahan: Penyedia yang ditentukan tidak ada.
- Kesalahan: Diminta DurationSeconds melebihi MaxSessionDuration set untuk peran ini.
- Kesalahan: Respons tidak berisi audiens yang diperlukan.

Kesalahan: Permintaan Anda menyertakan respons yang tidak validSAML. Untuk logout, klik di sini.

Kesalahan ini dapat terjadi ketika SAML respons dari penyedia identitas tidak menyertakan atribut dengan `Name` set ke`https://aws.amazon.com/SAML/Attributes/Role`. Atribut harus berisi satu atau beberapa elemen `AttributeValue`, masing-masing berisi sepasang string yang dipisahkan koma:

- ARNPeran yang dapat dipetakan oleh pengguna
- ARNSAMLPenyedia

Untuk informasi selengkapnya, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi. Untuk melihat SAML respons di browser Anda, ikuti langkah-langkah yang tercantum diLihat SAML respons di browser Anda.

Kesalahan: RoleSessionName diperlukan dalam AuthnResponse (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan: InvalidIdentityToken)

Kesalahan ini dapat terjadi ketika SAML respons dari penyedia identitas tidak menyertakan atribut dengan `Name` set ke`https://aws.amazon.com/SAML/Attributes/RoleSessionName`. Nilai atribut adalah pengidentifikasi untuk pengguna dan umumnya adalah ID pengguna atau alamat email.

Untuk informasi selengkapnya, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi. Untuk melihat SAML respons di browser Anda, ikuti langkah-langkah yang tercantum diLihat SAML respons di browser Anda.

Kesalahan: Tidak berwenang untuk melakukan sts: AssumeRoleWith SAML (layanan: AWSSecurityTokenService; kode status: 403; kode kesalahan: AccessDenied)

Kesalahan ini dapat terjadi jika IAM peran yang ditentukan dalam SAML respons salah eja atau tidak ada. Pastikan untuk menggunakan nama yang tepat dari peran Anda, karena nama peran bersifat peka kapitalisasi huruf. Perbaiki nama peran dalam konfigurasi penyedia SAML layanan.

Anda diperbolehkan mengakses hanya jika kebijakan kepercayaan peran Anda mencakup tindakan `sts:AssumeRoleWithSAML`. Jika SAML pernyataan Anda dikonfigurasi untuk menggunakan `PrincipalTag`atribut, kebijakan kepercayaan Anda juga harus menyertakan tindakan. `sts:TagSession` Untuk informasi selengkapnya tentang tanda sesi, lihat Lulus tag sesi di AWS STS.

Kesalahan ini dapat terjadi jika Anda tidak memiliki izin `sts:SetSourceIdentity` dalam kebijakan kepercayaan peran Anda. Jika SAML pernyataan Anda dikonfigurasi untuk menggunakan `SourceIdentity`atribut, maka kebijakan kepercayaan Anda juga harus menyertakan tindakan tersebut`sts:SetSourceIdentity`. Untuk informasi selengkapnya tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Kesalahan ini juga dapat terjadi jika pengguna gabungan tidak memiliki izin untuk mengambil peran tersebut. Peran harus memiliki kebijakan kepercayaan yang menentukan ARN penyedia IAM SAML identitas sebagai. `Principal` Peran ini juga berisi kondisi yang mengontrol pengguna mana yang dapat mengasumsikan peran tersebut. Pastikan bahwa pengguna Anda memenuhi persyaratan kondisi.

Kesalahan ini juga dapat terjadi jika SAML respon tidak termasuk yang `Subject` mengandung a`NameID`.

Untuk informasi selengkapnya, lihat Menetapkan Izin di AWS untuk Pengguna Gabungan dan Konfigurasikan SAML pernyataan untuk respons otentikasi. Untuk melihat SAML respons di browser Anda, ikuti langkah-langkah yang tercantum diLihat SAML respons di browser Anda.

Kesalahan: RoleSessionName di AuthnResponse harus cocok [A-za-Z_0-9+=, .@-] {2,64} (layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService) InvalidIdentityToken

Kesalahan ini dapat terjadi jika nilai atribut `RoleSessionName` terlalu panjang atau mengandung karakter yang tidak valid. Panjang valid maksimum adalah 64 karakter.

Untuk informasi selengkapnya, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi. Untuk melihat SAML respons di browser Anda, ikuti langkah-langkah yang tercantum diLihat SAML respons di browser Anda.

Kesalahan: Identitas Sumber harus cocok dengan [A-za-Z_0-9+=, .@-] {2,64} dan tidak dimulai dengan **`"aws:"`** (layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService) InvalidIdentityToken

Kesalahan ini dapat terjadi jika nilai atribut `sourceIdentity` terlalu panjang atau mengandung karakter yang tidak valid. Panjang valid maksimum adalah 64 karakter. Untuk informasi selengkapnya tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Untuk informasi selengkapnya tentang membuat SAML pernyataan, lihat. Konfigurasikan SAML pernyataan untuk respons otentikasi Untuk melihat SAML respons di browser Anda, ikuti langkah-langkah yang tercantum diLihat SAML respons di browser Anda.

Kesalahan: Tanda tangan respons tidak valid (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan:) InvalidIdentityToken

Kesalahan ini dapat terjadi ketika metadata federasi penyedia identitas tidak cocok dengan metadata penyedia identitas. IAM Misalnya, file metadata untuk penyedia layanan identitas mungkin telah berubah untuk memperbarui sertifikat yang kedaluwarsa. Unduh file SAML metadata yang diperbarui dari penyedia layanan identitas Anda. Kemudian perbarui di entitas penyedia AWS identitas yang Anda tentukan IAM dengan CLI perintah `aws iam update-saml-provider` lintas platform atau `Update-IAMSAMLProvider` PowerShell cmdlet.

Kesalahan: Gagal mengambil peran: Penerbit tidak hadir di penyedia tertentu (layanan: AWSOpenIdDiscoveryService; kode status: 400; kode kesalahan: AuthSamlInvalidSamlResponseException)

Kesalahan ini dapat terjadi jika penerbit dalam SAML tanggapan tidak cocok dengan penerbit yang dinyatakan dalam file metadata federasi. File metadata diunggah AWS saat Anda membuat penyedia identitas di. IAM

Kesalahan: Tidak dapat mengurai metadata.

Kesalahan ini dapat terjadi jika Anda tidak memformat file metadata Anda dengan benar.

Saat Anda membuat atau mengelola penyedia SAML identitas di AWS Management Console, Anda harus mengambil dokumen SAML metadata dari penyedia identitas Anda.

File metadata ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons SAML otentikasi (pernyataan) yang diterima dari iDP. File metadata harus dikodekan dalam format UTF -8 tanpa tanda urutan byte (). BOM Untuk menghapusnyaBOM, Anda dapat menyandikan file sebagai UTF -8 menggunakan alat pengeditan teks, seperti Notepad + +.

Sertifikat x.509 yang disertakan sebagai bagian dari dokumen SAML metadata harus menggunakan ukuran kunci minimal 1024 bit. Selain itu, sertifikat x.509 juga harus bebas dari ekstensi berulang. Anda dapat menggunakan ekstensi, tetapi ekstensi hanya dapat muncul sekali dalam sertifikat. Jika sertifikat x.509 tidak memenuhi salah satu kondisi, pembuatan IDP gagal dan mengembalikan kesalahan “Tidak dapat mengurai metadata”.

Seperti yang didefinisikan oleh Profil Interoperabilitas Metadata SAML V2.0 Versi 1.0, IAM tidak mengevaluasi atau mengambil tindakan terkait berakhirnya sertifikat X.509 dokumen metadata.

Kesalahan: Penyedia yang ditentukan tidak ada.

Kesalahan ini dapat terjadi jika nama penyedia dalam SAML pernyataan tidak cocok dengan nama penyedia di. IAM Untuk informasi selengkapnya tentang melihat nama penyedia, lihat Buat penyedia SAML identitas di IAM.

Kesalahan: Diminta DurationSeconds melebihi MaxSessionDuration set untuk peran ini.

Kesalahan ini dapat terjadi jika Anda mengambil peran dari AWS CLI atauAPI.

Saat Anda menggunakan AssumeRoleWithSAMLAPIoperasi assume-role-with-samlCLIatau untuk mengambil peran, Anda dapat menentukan nilai untuk `DurationSeconds` parameter tersebut. Anda dapat menentukan nilai dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Jika Anda menentukan nilai yang lebih tinggi dari pengaturan ini, operasi gagal. Misalnya, jika Anda menentukan durasi sesi 12 jam, tetapi administrator Anda mengatur durasi sesi maksimum menjadi 6 jam, operasi Anda gagal. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Kesalahan: Respons tidak berisi audiens yang diperlukan.

Kesalahan ini dapat terjadi jika ada ketidakcocokan antara audiens URL dan penyedia identitas dalam SAML konfigurasi. Pastikan bahwa pengenal pihak yang mengandalkan penyedia identitas (IDP) Anda sama persis dengan audiens URL (ID entitas) yang disediakan dalam konfigurasi. SAML

Bagaimana IAM bekerja dengan AWS layanan lain

Meskipun IAM merupakan AWS layanan utama yang akan Anda gunakan untuk mengelola IAM sumber daya, semua AWS layanan lain bekerja dengan IAM untuk mengontrol akses ke sumber daya di akun Anda.

- AWS CloudFormation

  AWS CloudFormation terintegrasi dengan IAM dengan memungkinkan Anda untuk mendefinisikan dan mengelola IAM sumber daya sebagai bagian dari AWS CloudFormation template Anda.Anda dapat menggunakan AWS CloudFormation untuk menentukan IAM izin yang diperlukan untuk sumber daya lain yang Anda berikan. AWS AWS CloudFormation juga mendukung penggunaan IAM peran untuk mengelola kredensional yang diperlukan untuk penyediaan dan pengelolaan AWS infrastruktur Anda, dan fitur deteksi driftnya membantu Anda menjaga integritas konfigurasi Anda. IAM
- AWS CloudShell

  Ketika Anda mengakses AWS CloudShell, otentikasi dan otorisasi Anda ditangani melalui. IAM AWS CloudShell berjalan dalam konteks IAM peran yang ditetapkan ke pengguna atau akun Anda. Saat Anda meluncurkan AWS CloudShell, secara otomatis menghasilkan kredenal keamanan sementara berdasarkan IAM peran yang ditetapkan kepada Anda.
- AWS SDKs

  AWS SDKsPekerjaan IAM dengan menangani proses otentikasi dan otorisasi, mengelola AWS kredensional, dan menghormati izin dan kebijakan yang ditetapkan IAM untuk memastikan aplikasi Anda hanya dapat mengakses sumber daya yang diizinkan untuk digunakan. SDKsMenyediakan mekanisme untuk memperoleh dan menggunakan kredenal keamanan sementara, serta memvalidasi izin yang diperlukan untuk operasi aplikasi Anda.

Untuk daftar AWS layanan yang bekerja dengan IAM dan IAM fitur dukungan layanan, lihatAWS layanan yang bekerja dengan IAM.

Buat sumber daya IAM dengan AWS CloudFormation

AWS Identity and Access Management terintegrasi dengan AWS CloudFormation, layanan yang membantu Anda memodelkan dan mengatur AWS sumber daya Anda sehingga Anda dapat menghabiskan lebih sedikit waktu untuk membuat dan mengelola sumber daya dan infrastruktur

Anda. Anda membuat templat yang menjelaskan semua AWS sumber daya yang Anda inginkan (seperti kunci akses, grup, kebijakan grup, profil instans, kebijakan terkelola, penyedia OIDC, kebijakan sebaris, peran, kebijakan peran, penyedia SAMM, sertifikat server, peran terkait layanan, pengguna (dan menambahkan pengguna ke grup), kebijakan pengguna, dan perangkat MFA virtual), serta menyediakan serta mengonfigurasi sumber daya tersebut untuk Anda. AWS CloudFormation

Bila Anda menggunakan AWS CloudFormation, Anda dapat menggunakan kembali template Anda untuk mengatur sumber daya IAM Anda secara konsisten dan berulang kali. Jelaskan sumber daya Anda sekali, lalu sediakan sumber daya yang sama berulang-ulang di beberapa Akun AWS dan Wilayah.

IAM dan template AWS CloudFormation

Untuk menyediakan dan mengonfigurasi sumber daya untuk IAM dan layanan terkait, Anda harus memahami AWS CloudFormation templat. Templat adalah file teks dengan format JSON atau YAML. Template ini menjelaskan sumber daya yang ingin Anda sediakan di AWS CloudFormation tumpukan Anda. Jika Anda tidak terbiasa dengan JSON atau YAMM, Anda dapat menggunakan AWS CloudFormation Designer untuk membantu Anda memulai dengan template. AWS CloudFormation Untuk informasi selengkapnya, lihat Apa itu AWS CloudFormation Designer? di Panduan Pengguna AWS CloudFormation .

IAM mendukung pembuatan kunci akses, grup, kebijakan grup, profil instans, kebijakan terkelola, penyedia OIDC, kebijakan inline, peran, kebijakan peran, penyedia SAMM, sertifikat server, peran terkait layanan, pengguna (dan menambahkan pengguna ke grup), kebijakan pengguna, dan perangkat MFA virtual di. AWS CloudFormationUntuk informasi selengkapnya, termasuk contoh template JSON dan YAMAL untuk sumber daya IAM, lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan Pengguna.AWS CloudFormation

Anda juga dapat membuat templat yang membuat sumber daya terkait, seperti peran dan kebijakan terkelola.

Pelajari lebih lanjut tentang AWS CloudFormation

Untuk mempelajari selengkapnya AWS CloudFormation, lihat sumber daya berikut:

- AWS CloudFormation
- AWS CloudFormation Panduan Pengguna
- AWS CloudFormation Referensi API
- AWS CloudFormation Panduan Pengguna Antarmuka Baris Perintah

Gunakan AWS CloudShell untuk bekerja dengan AWS Identity and Access Management

AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console Anda dapat menjalankan AWS CLI perintah terhadap AWS layanan (termasuk AWS Identity and Access Management) menggunakan shell pilihan Anda (Bash, PowerShell atau Z shell). Anda juga dapat melakukan ini tanpa perlu mengunduh atau menginstal alat baris perintah.

Anda meluncurkan AWS CloudShell dari AWS Management Console, dan AWS kredenal yang Anda gunakan untuk masuk ke konsol secara otomatis tersedia di sesi shell baru. Pra-otentikasi AWS CloudShell pengguna ini memungkinkan Anda untuk melewati konfigurasi kredensional saat berinteraksi dengan AWS layanan seperti IAM menggunakan AWS CLI versi 2 (pra-instal pada lingkungan komputasi shell).

Dapatkan izin IAM untuk AWS CloudShell

Dengan menggunakan sumber daya manajemen akses yang disediakan oleh AWS Identity and Access Management, administrator dapat memberikan izin kepada pengguna IAM sehingga mereka dapat mengakses AWS CloudShell dan menggunakan fitur lingkungan.

Cara tercepat bagi administrator untuk memberikan akses ke pengguna adalah melalui kebijakan AWS terkelola. Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Kebijakan AWS terkelola berikut ini CloudShell dapat dilampirkan ke identitas IAM:

- `AWSCloudShellFullAccess`: Memberikan izin untuk menggunakan AWS CloudShell dengan akses penuh ke semua fitur.

Jika ingin membatasi cakupan tindakan yang dapat dilakukan oleh pengguna IAM AWS CloudShell, Anda dapat membuat kebijakan kustom yang menggunakan kebijakan `AWSCloudShellFullAccess` terkelola sebagai templat. Untuk informasi selengkapnya tentang membatasi tindakan yang tersedia bagi pengguna CloudShell, lihat Mengelola AWS CloudShell akses dan penggunaan dengan kebijakan IAM di Panduan AWS CloudShell Pengguna.

Berinteraksi dengan IAM

Setelah Anda meluncurkan AWS CloudShell dari AWS Management Console, Anda dapat segera mulai berinteraksi dengan IAM menggunakan antarmuka baris perintah.

> **Note**
>
> Saat menggunakan AWS CLI in AWS CloudShell, Anda tidak perlu mengunduh atau menginstal sumber daya tambahan apa pun. Selain itu, karena Anda sudah diautentikasi di dalam shell, Anda tidak perlu mengonfigurasi kredensional sebelum melakukan panggilan.

Buat grup IAM dan tambahkan pengguna IAM ke grup menggunakan AWS CloudShell

Contoh berikut digunakan CloudShell untuk membuat grup IAM, menambahkan pengguna IAM ke grup, dan kemudian memverifikasi bahwa perintah berhasil.

1. Dari AWS Management Console, Anda dapat meluncurkan CloudShell dengan memilih opsi berikut yang tersedia di bilah navigasi:
   - Pilih CloudShell ikonnya.
   - Mulai mengetik “cloudshell” di kotak Pencarian dan kemudian pilih opsi. CloudShell
2. Untuk membuat grup IAM, masukkan perintah berikut di baris CloudShell perintah. Dalam contoh ini kami menamai grup east_coast:

   ```
   aws iam create-group --group-name east_coast
   ```

   Jika panggilan berhasil, baris perintah menampilkan respons dari layanan yang mirip dengan output berikut:

   ```
   {
      "Group": {
          "Path": "/",
          "GroupName": "east_coast",
          "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
          "Arn": "arn:aws:iam::111122223333:group/east_coast",
          "CreateDate": "2023-09-11T21:02:21+00:00"
       }
   }
   ```

3. Untuk menambahkan pengguna ke grup yang Anda buat, gunakan perintah berikut, tentukan nama grup dan nama pengguna. Dalam contoh ini kami menamai grup east_coast dan pengguna johndoe:

```
aws iam add-user-to-group --group-name east_coast --user-name johndoe
```

4. Untuk memverifikasi bahwa pengguna berada dalam grup, gunakan perintah berikut, tentukan nama grup. Dalam contoh ini kita terus menggunakan grup east_coast:

```
aws iam get-group --group-name east_coast
```

Jika panggilan berhasil, baris perintah menampilkan respons dari layanan yang mirip dengan output berikut:

```
      {
        "Users": [
          {
              "Path": "/",
              "UserName": "johndoe",
              "UserId": "AIDAYBDBW4JBXGEXAMPLE",
              "Arn": "arn:aws:iam::552108220995:user/johndoe",
              "CreateDate": "2023-09-11T20:43:14+00:00",
              "PasswordLastUsed": "2023-09-11T20:59:14+00:00"
          }
        ],
        "Group": {
              "Path": "/",
              "GroupName": "east_coast",
              "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
              "Arn": "arn:aws:iam::111122223333:group/east_coast",
              "CreateDate": "2023-09-11T21:02:21+00:00"
           }
       }
```

Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDKs) tersedia untuk banyak bahasa pemrograman populer. Masing-masing SDK menyediakanAPI, contoh kode, dan dokumentasi yang memudahkan pengembang untuk membangun aplikasi dalam bahasa pilihan mereka.

SDKdokumentasi	Contoh kode
AWS SDK for C++	AWS SDK for C++ contoh kode
AWS CLI	AWS CLI contoh kode
AWS SDK for Go	AWS SDK for Go contoh kode
AWS SDK for Java	AWS SDK for Java contoh kode
AWS SDK for JavaScript	AWS SDK for JavaScript contoh kode
AWS SDK for Kotlin	AWS SDK for Kotlin contoh kode
AWS SDK for .NET	AWS SDK for .NET contoh kode
AWS SDK for PHP	AWS SDK for PHP contoh kode
AWS Tools for PowerShell	Alat untuk contoh PowerShell kode
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) contoh kode
AWS SDK for Ruby	AWS SDK for Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK untuk SAP ABAP	AWS SDK untuk SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Untuk contoh khusus untuk layanan ini, lihat Contoh kode untuk IAM menggunakan AWS SDKs.

Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

Informasi referensi untuk AWS Identity and Access Management

Gunakan topik di bagian ini untuk menemukan bahan referensi terperinci untuk berbagai aspek IAM dan AWS STS.

Topik

Identifikasi AWS sumber daya dengan Nama Sumber Daya Amazon (ARNs)

Amazon Resource Names (ARNs) mengidentifikasi AWS sumber daya secara unik. Kami memerlukan ARN kapan Anda perlu menentukan sumber daya secara jelas di semua AWS, seperti dalam IAM kebijakan, tag Amazon Relational Database Service (AmazonRDS), dan panggilan. API

ARNformat

Berikut ini adalah format umum untukARNs. Format spesifik bergantung pada sumber daya. Untuk menggunakanARN, ganti *italicized* teks dengan informasi khusus sumber daya. Ketahuilah bahwa ARNs untuk beberapa sumber daya menghilangkan Wilayah, ID akun, atau Wilayah dan ID akun.

```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partisi

Partisi tempat sumber daya berada. Partisi adalah sekelompok AWS Wilayah. Setiap AWS akun dicakup ke satu partisi.

Berikut ini adalah partisi yang didukung:

- `aws`- AWS Wilayah
- `aws-cn` - Wilayah Tiongkok
- `aws-us-gov` - AWS GovCloud (US) Wilayah

layanan

Namespace layanan yang mengidentifikasi produk. AWS

region

Kode Wilayah. Misalnya, `us-east-2` untuk US East (Ohio). Untuk daftar kode Region, lihat Titik akhir Regional di. Referensi Umum AWS

account-id

ID AWS akun yang memiliki sumber daya, tanpa tanda hubung. Misalnya, `123456789012`.

jenis sumber daya

Jenis sumber daya . Misalnya, `vpc` untuk cloud pribadi virtual (VPC).

resource-id

Pengidentifikasi sumber daya. Ini adalah nama sumber daya, ID sumber daya, atau jalur sumber daya. Beberapa pengidentifikasi sumber daya menyertakan sumber daya induk (sub-resource-type/parent-resource/sub-resource) atau qualifier seperti versi (resource-type:resource-name:qualifier).

Contoh

IAMpengguna

arn:aws:iam::*123456789012*:pengguna/*johndoe*

SNStopik

arn:aws:sns:*us-east-1*:*123456789012*:*example-sns-topic-name*

VPC

arn:aws:ec2:*us-east-1*:*123456789012*:vpc/*vpc-0e9801d129EXAMPLE*

Cari ARN format untuk sumber daya

Format yang tepat ARN tergantung pada jenis layanan dan sumber daya. Beberapa sumber daya ARNs dapat mencakup jalur, variabel, atau wildcard. Untuk mencari ARN format AWS sumber daya tertentu, buka Referensi Otorisasi Layanan, buka halaman untuk layanan, dan arahkan ke tabel jenis sumber daya.

Jalur di ARNs

Sumber daya ARNs dapat mencakup jalur. Misalnya, di Amazon S3, pengidentifikasi sumber daya adalah nama objek yang dapat menyertakan garis miring maju (/) untuk membentuk jalur. Demikian pula, nama IAM pengguna dan nama grup dapat menyertakan jalur. Hanya karakter alfanumerik dan karakter berikut yang diizinkan di IAM jalur: garis miring maju (/), plus (), sama dengan (+), koma (=), periode (,), at (), garis bawah (.@), dan tanda hubung (_). -

Menggunakan wildcard di jalur

Jalur dapat menyertakan karakter wildcard, yaitu tanda bintang (*). Misalnya, jika Anda menulis IAM kebijakan, Anda dapat menentukan semua IAM pengguna yang memiliki jalur `product_1234` menggunakan wildcard sebagai berikut:

```
arn:aws:iam::123456789012:user/Development/product_1234/*
```

Demikian pula, Anda dapat menentukan `user/*` yang berarti semua pengguna atau `group/*` yang berarti semua grup, seperti dalam contoh berikut:

```
"Resource":"arn:aws:iam::123456789012:user/*"
"Resource":"arn:aws:iam::123456789012:group/*"
```

Contoh berikut menunjukkan ARNs bucket Amazon S3 di mana nama sumber daya menyertakan jalur:

```
arn:aws:s3:::my_corporate_bucket/*
arn:aws:s3:::my-corporate-bucket/Development/*
```

Penggunaan wildcard salah

Anda tidak dapat menggunakan wildcard di bagian ARN yang menentukan jenis sumber daya, seperti istilah `user` dalam. IAM ARN Misalnya, hal berikut tidak diizinkan.

```
arn:aws:iam::123456789012:u*   <== not allowed
```

IAMpengidentifikasi

IAMmenggunakan beberapa pengidentifikasi yang berbeda untuk pengguna, IAM grup, peran, kebijakan, dan sertifikat server. Bagian ini menjelaskan pengidentifikasi dan kapan Anda menggunakan masing-masing pengidentifikasi.

Topik

- Nama dan jalur yang ramah
- IAM ARNs
- Pengidentifikasi unik

Nama dan jalur yang ramah

Saat Anda membuat pengguna, peran, grup pengguna, atau kebijakan, atau saat Anda mengunggah sertifikat server, Anda memberikan nama yang ramah. Contohnya termasuk Bob, TestApp 1, Pengembang, ManageCredentialsPermissions, atau ProdServerCert.

Jika Anda menggunakan IAM API or AWS Command Line Interface (AWS CLI) untuk membuat IAM sumber daya, Anda dapat menambahkan jalur opsional. Anda dapat menggunakan satu jalur, atau sarang beberapa jalur sebagai struktur folder. Misalnya, Anda dapat menggunakan jalur bersarang `/division_abc/subdivision_xyz/product_1234/engineering/` untuk mencocokkan struktur organisasi perusahaan Anda. Anda kemudian dapat membuat kebijakan untuk mengizinkan semua pengguna di jalur tersebut mengakses simulator kebijakanAPI. Untuk melihat kebijakan ini, lihat IAM: Akses API simulator kebijakan berdasarkan jalur pengguna. Untuk informasi tentang bagaimana nama ramah dapat ditentukan, lihat APIdokumentasi Pengguna. Untuk contoh-contoh tambahan tentang bagaimana Anda mungkin menggunakan jalur, lihat IAM ARNs.

Saat digunakan AWS CloudFormation untuk membuat sumber daya, Anda dapat menentukan jalur untuk pengguna, IAM grup, dan peran, serta kebijakan yang dikelola pelanggan.

Jika Anda memiliki pengguna dan grup pengguna di jalur yang sama, IAM tidak secara otomatis menempatkan pengguna dalam grup pengguna tersebut. Misalnya, Anda dapat membuat grup pengguna Pengembang dan menentukan jalurnya sebagai`/division_abc/subdivision_xyz/product_1234/engineering/`. Jika Anda membuat pengguna bernama Bob dan menambahkan jalur yang sama kepadanya, ini tidak secara otomatis menempatkan Bob di grup pengguna Pengembang. IAMtidak memberlakukan batasan apa pun antara pengguna atau IAM grup berdasarkan jalur mereka. Pengguna dengan jalur yang berbeda dapat menggunakan sumber daya yang sama jika mereka telah diberikan izin untuk sumber daya tersebut. Jumlah dan ukuran IAM sumber daya dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota.

IAM ARNs

Sebagian besar sumber daya memiliki nama yang ramah misalnya, nama pengguna Bob atau grup pengguna bernama`Developers`. Namun, bahasa kebijakan izin mengharuskan Anda menentukan sumber daya atau sumber daya menggunakan format Amazon Resource Name (ARN) berikut.

```
arn:partition:service:region:account:resource
```

Di mana:

- `partition`mengidentifikasi partisi untuk sumber daya. Untuk Wilayah AWS standar, partisinya adalah `aws`. Jika Anda memiliki sumber daya di partisi lain, maka partisi-nya adalah `aws-`*`partitionname`*. Contohnya, partisi untuk sumber daya di Wilayah Tiongkok (Beijing) adalah `aws-cn`. Anda tidak dapat mendelegasikan akses di antara akun di partisi yang berbeda.
- `service`mengidentifikasi AWS produk. IAMsumber daya selalu digunakan`iam`.
- `region`mengidentifikasi Wilayah sumber daya. Untuk IAM sumber daya, ini selalu dikosongkan.
- `account`menentukan Akun AWS ID tanpa tanda hubung.
- `resource`mengidentifikasi sumber daya tertentu dengan nama.

Anda dapat menentukan IAM dan AWS STS ARNs menggunakan sintaks berikut. Bagian Wilayah kosong karena IAM sumber daya bersifat global. ARN

Sintaksis:

```
arn:aws:iam::account:root
```

```
arn:aws:iam::account:user/user-name-with-path
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:sts::account:self
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
```

Banyak dari contoh berikut termasuk jalur di bagian sumber dayaARN. Jalur tidak dapat dibuat atau dimanipulasi di AWS Management Console. Untuk menggunakan jalur, Anda harus bekerja dengan sumber daya dengan menggunakan AWS CLI,, atau Alat untuk Windows PowerShell. AWS API

Contoh:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/JohnDoe
arn:aws:sts::123456789012:assumed-role/Accounting-Role/JaneDoe
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneDoeMFA
arn:aws:iam::123456789012:u2f/user/JohnDoe/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
```

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

Contoh berikut memberikan detail lebih lanjut untuk membantu Anda memahami ARN format untuk berbagai jenis IAM dan AWS STS sumber daya.

- Pengguna IAM di akun:

> **Note**
>
> Setiap nama IAM pengguna adalah unik. Nama pengguna tidak peka huruf besar/kecil untuk pengguna, seperti selama proses masuk, tetapi peka huruf besar/kecil saat Anda menggunakannya dalam kebijakan atau sebagai bagian dari. ARN

```
arn:aws:iam::123456789012:user/JohnDoe
```

- Pengguna lain dengan jalur yang mencerminkan bagan organisasi:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
```

- Grup pengguna IAM:

```
arn:aws:iam::123456789012:group/Developers
```

- Grup IAM pengguna dengan jalur:

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- Peran IAM:

```
arn:aws:iam::123456789012:role/S3Access
```

- Peran yang terhubung dengan layanan:

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- Peran layanan:

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- Kebijakan terkelola:

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- Profil instans yang dapat dikaitkan dengan EC2 instans Amazon:

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- Pengguna federasi yang diidentifikasi IAM sebagai “Paulo”:

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- Sesi aktif seseorang yang mengambil peran "Akuntansi-Peran", dengan nama sesi peran "Mary":

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- Merupakan sesi pemanggil sendiri saat digunakan sebagai sumber daya dalam API panggilan, seperti AWS STS SetContextAPI, yang beroperasi pada sesi panggilan:

```
arn:aws:sts::123456789012:self
```

- Perangkat autentikasi multi-faktor yang ditetapkan ke pengguna bernama Jorge:

```
arn:aws:iam::123456789012:mfa/Jorge
```

- Sertifikat server:

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- Sertifikat server dengan jalur yang mencerminkan bagan organisasi:

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- Penyedia identitas (SAMLdanOIDC):

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
```

```
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- OIDCpenyedia identitas dengan jalur yang mencerminkan penyedia EKS OIDC identitas AmazonURL:

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

Penting lainnya ARN adalah pengguna rootARN. Meskipun ini bukan IAM sumber daya, Anda harus terbiasa dengan format iniARN. Ini sering digunakan dalam `Principal`elemen kebijakan berbasis sumber daya.

- Akun AWS Menampilkan yang berikut ini:

```
arn:aws:iam::123456789012:root
```

Contoh berikut menunjukkan kebijakan yang dapat Anda tetapkan kepada Richard untuk mengizinkannya mengelola kunci aksesnya. Perhatikan bahwa sumber daya adalah IAM pengguna Richard.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/Richard"
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
```

```
}
```

> **Note**
>
> Saat Anda menggunakan ARNs untuk mengidentifikasi sumber daya dalam IAM kebijakan, Anda dapat menyertakan variabel kebijakan. Variabel kebijakan dapat menyertakan placeholder untuk informasi runtime (seperti nama pengguna) sebagai bagian dari. ARN Untuk informasi selengkapnya, silakan lihat Elemen kebijakan IAM: Variabel dan tanda

Menggunakan wildcard dan path di ARNs

Anda dapat menggunakan wildcard di *`resource`* bagian dari ARN untuk menentukan beberapa pengguna atau IAM grup atau kebijakan. Misalnya, untuk menentukan semua pengguna yang bekerja pada product_1234, Anda menggunakan:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Jika Anda memiliki pengguna yang namanya dimulai dengan stringapp_, Anda dapat merujuk mereka semua dengan yang berikut iniARN.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Untuk menentukan semua pengguna, IAM grup, atau kebijakan dalam Anda Akun AWS, gunakan wildcard setelah `user/group/`, atau `policy/` bagian dariARN, masing-masing.

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Jika Anda menentukan berikut ini ARN untuk penggunaan`arn:aws:iam::111122223333:user/*`, itu cocok dengan kedua contoh berikut.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Tetapi, jika Anda menentukan yang berikut ARN untuk pengguna `arn:aws:iam::111122223333:user/division_abc*` itu cocok dengan contoh kedua, tetapi bukan yang pertama.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Jangan gunakan wildcard di`user/`,`group/`, atau `policy/` bagian dari. ARN Misalnya, IAM tidak mengizinkan yang berikut:

```
arn:aws:iam::123456789012:u*
```

Example Contoh penggunaan jalur dan ARNs untuk grup pengguna berbasis proyek

Jalur tidak dapat dibuat atau dimanipulasi di AWS Management Console. Untuk menggunakan jalur, Anda harus bekerja dengan sumber daya dengan menggunakan AWS API, the AWS CLI, atau Tools untuk Windows PowerShell.

Dalam contoh ini, Jules dalam grup pengguna Marketing_Admin membuat grup pengguna berbasis proyek dalam jalur /marketing/. Jules menugaskan pengguna dari berbagai bagian perusahaan ke grup pengguna. Contoh ini menggambarkan bahwa jalur pengguna tidak terkait dengan grup pengguna yang pengguna ikuti.

Grup pemasaran memiliki produk baru yang akan mereka luncurkan, sehingga Jules membuat grup pengguna baru dalam jalur /marketing/ yang disebut Widget_Launch. Jules kemudian menetapkan kebijakan berikut kepada grup pengguna, yang memberi grup pengguna akses ke objek di bagian `example_bucket` yang ditetapkan untuk peluncuran khusus ini.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/widget/*"}}
    }
  ]
}
```

Jules, kemudian menugaskan pengguna yang mengerjakan peluncuran ini ke grup pengguna. Ini termasuk Patricia dan Eli dari jalur /pemasaran/. Ini juga mencakup Chris dan Chloe dari jalur / pemasaran/, dan Alice serta Jim dari /legal/.

Pengidentifikasi unik

Saat IAM membuat pengguna, grup pengguna, peran, kebijakan, profil instance, atau sertifikat server, ia menetapkan ID unik untuk setiap sumber daya. ID unik terlihat seperti ini:

```
AIDAJQABLZS4A3QDU576Q
```

Untuk sebagian besar, Anda menggunakan nama yang ramah dan ARNsketika Anda bekerja dengan IAM sumber daya. Dengan begitu, Anda tidak perlu mengetahui ID unik untuk sumber daya tertentu. Namun, ID unik terkadang dapat berguna jika saat situasi tidak praktis untuk menggunakan nama yang ramah.

Salah satu contoh menggunakan kembali nama ramah di Anda Akun AWS. Dalam akun Anda, nama ramah untuk pengguna, grup pengguna, peran, atau kebijakan harus unik. Misalnya, Anda dapat membuat nama IAM penggunaJohn. Perusahaan Anda menggunakan Amazon S3 dan memiliki folder dengan bucket untuk setiap karyawan. IAMuser John adalah anggota grup IAM pengguna bernama User-S3-Access dengan izin yang memungkinkan pengguna mengakses hanya ke folder mereka sendiri di bucket. Untuk contoh cara membuat kebijakan berbasis identitas yang memungkinkan IAM pengguna mengakses objek bucket mereka sendiri di S3 menggunakan nama pengguna yang ramah, lihat. Amazon S3: Memungkinkan IAM pengguna mengakses direktori home S3 mereka, secara terprogram dan di konsol

Misalkan karyawan bernama John meninggalkan perusahaan Anda dan Anda menghapus nama IAM pengguna yang sesuaiJohn. Tapi kemudian karyawan lain bernama John mulai, dan Anda membuat IAM pengguna baru bernamaJohn. Anda menambahkan nama IAM pengguna baru John ke grup IAM pengguna yang adaUser-S3-Access. Jika kebijakan yang terkait dengan grup pengguna menentukan nama IAM pengguna yang ramahJohn, kebijakan tersebut mengizinkan John baru mengakses informasi yang ditinggalkan oleh John sebelumnya.

Secara umum, kami menyarankan Anda menentukan ARN sumber daya dalam kebijakan Anda, bukan ID uniknya. Namun, setiap IAM pengguna memiliki ID unik, bahkan jika Anda membuat IAM pengguna baru yang menggunakan kembali nama ramah yang Anda hapus sebelumnya. Dalam contoh, IAM pengguna lama John dan IAM pengguna baru John memiliki keunikan yang berbedaIDs. Anda dapat membuat kebijakan berbasis sumber daya yang memberikan akses dengan ID unik dan tidak hanya dengan nama pengguna. Melakukannya mengurangi kemungkinan bahwa

Anda secara tidak sengaja dapat memberikan akses ke informasi yang seharusnya tidak dimiliki oleh seorang karyawan.

Contoh berikut menunjukkan bagaimana Anda dapat menentukan unik IDs dalam `Principal`elemen kebijakan berbasis sumber daya.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

Contoh berikut menunjukkan bagaimana Anda dapat menentukan unik IDs dalam `Condition`elemen kebijakan menggunakan kunci kondisi global `aws:userid`.

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

Contoh lain di mana pengguna IDs dapat berguna adalah jika Anda memelihara database Anda sendiri (atau toko lain) informasi IAM pengguna atau peran. ID unik dapat memberikan pengenal unik untuk setiap IAM pengguna atau peran yang Anda buat. Ini adalah kasus ketika Anda memiliki IAM pengguna atau peran yang menggunakan kembali nama, seperti pada contoh sebelumnya.

Memahami prefiks ID yang unik

IAMmenggunakan awalan berikut untuk menunjukkan jenis sumber daya apa yang berlaku untuk setiap ID unik. Awalan dapat bervariasi berdasarkan kapan mereka dibuat.

Awalan	Jenis sumber daya
ABIA	AWS STS token pembawa layanan

Awalan	Jenis sumber daya
ACCA	Kredensial spesifik konteks
AGPA	Grup pengguna
AIDA	IAMpengguna
AIPA	Profil EC2 contoh Amazon
AKIA	Kunci akses
ANPA	Kebijakan terkelola
ANVA	Versi dalam kebijakan terkelola
APKA	Kunci publik
AROA	Peran
ASCA	Sertifikat
ASIA	Kunci akses sementara (AWS STS) IDs menggunakan awalan ini, tetapi unik hanya dalam kombinasi dengan kunci akses rahasia dan token sesi.

Mendapatkan pengidentifikasi unik

ID unik untuk IAM sumber daya tidak tersedia di IAM konsol. Untuk mendapatkan ID unik, Anda dapat menggunakan AWS CLI perintah atau IAM API panggilan berikut.

AWS CLI:

- get-caller-identity
- dapatkan-kelompok
- dapatkan-peran
- dapatkan-pengguna
- dapatkan-kebijakan

- get-instance-profile
- get-server-certificate

IAM API:

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAMdan AWS STS kuota

AWS Identity and Access Management (IAM) dan AWS Security Token Service (STS) memiliki kuota yang membatasi ukuran objek. Hal ini memengaruhi cara Anda menamai sebuah objek, jumlah objek yang dapat Anda buat, dan jumlah karakter yang dapat Anda gunakan saat Anda melewati sebuah objek.

> **Note**
>
> Untuk mendapatkan informasi tingkat akun tentang IAM penggunaan dan kuota, gunakan GetAccountSummaryAPIoperasi atau perintah. get-account-summary AWS CLI

Persyaratan nama IAM

IAMnama memiliki persyaratan dan batasan berikut:

- Dokumen kebijakan hanya dapat berisi karakter Unicode berikut: tab horizontal (U+0009), umpan baris (U+000A), pengembalian pengangkutan (U+000D), dan karakter dalam rentang U+0020 hingga U+00FF.
- Nama pengguna, grup, peran, kebijakan, profil instans, sertifikat server, dan jalur harus alfanumerik, termasuk karakter umum berikut: plus (+), sama dengan (=), koma (,), periode (.), at

(@), garis bawah (_), dan tanda hubung (-). Nama jalur harus dimulai dan diakhiri dengan garis miring ke depan (/).

- Nama pengguna, grup, peran, dan profil instans harus unik dalam akun. Mereka tidak dibedakan berdasarkan kasus, misalnya, Anda tidak dapat membuat grup bernama keduanya **`ADMINS`** dan**`admins`**.
- Nilai ID eksternal yang digunakan pihak ketiga untuk mengambil peran harus memiliki minimal 2 karakter dan maksimal 1.224 karakter. Nilai harus berupa alfanumerik tanpa spasi. Nilai dapat mencakup simbol berikut: plus (+), setara (=), koma (,), titik (.), a keong (@), titik dua (:), garis miring (/), dan tanda hubung (-). Untuk informasi selengkapnya tentang ID eksternal, lihat Akses ke Akun AWS yang dimiliki oleh pihak ketiga.
- Nama kebijakan untuk kebijakan sebaris harus unik untuk pengguna, grup, atau peran yang disematkan. Nama-nama dapat berisi karakter Latin Dasar (ASCII) kecuali untuk karakter cadangan berikut: garis miring mundur (\), garis miring maju (/), tanda bintang (*), tanda tanya (?) dan ruang putih. Karakter-karakter ini dicadangkan menurut RFC3986, bagian 2.2.
- Kata sandi pengguna (profil login) dapat berisi karakter Latin Dasar (ASCII) apa pun.
- Akun AWS Alias ID harus unik di seluruh AWS produk, dan harus alfanumerik mengikuti konvensi penamaan. DNS Alias harus huruf kecil, tidak boleh dimulai atau diakhiri dengan tanda hubung, tidak dapat berisi dua tanda hubung berturut-turut, dan tidak bisa berupa angka 12 digit.

Untuk daftar karakter Latin Dasar (ASCII), buka Tabel Kode Library of Congress Basic Latin (ASCII).

Kuota objek IAM

Kuota, juga disebut sebagai batas dalam AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item dalam Anda Akun AWS. Gunakan Service Quotas untuk mengelola kuota AndaIAM.

Untuk daftar titik akhir IAM layanan dan kuota layanan, lihat AWS Identity and Access Management titik akhir dan kuota di. Referensi Umum AWS

Untuk meminta kenaikan kuota

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk untuk masuk ke. AWS Management Console
2. Buka Konsol Service Quotas.
3. Di panel navigasi, pilih Layanan AWS .

4. Di bilah navigasi, pilih Wilayah AS Timur (N. Virginia). Lalu cari **IAM**.
5. Pilih AWS Identity and Access Management (IAM), pilih kuota, dan ikuti petunjuk untuk meminta kenaikan kuota.

Untuk informasi selengkapnya, lihat Meminta Peningkatan Kuota dalam Panduan Pengguna Service Quotas.

Untuk melihat contoh cara meminta peningkatan IAM kuota menggunakan konsol Service Quotas, tonton video berikut.

Minta peningkatan IAM kuota menggunakan konsol Service Quotas.

Anda dapat meminta peningkatan kuota default untuk kuota yang dapat disesuaikanIAM. Permintaan hingga permintaan maximum quota secara otomatis disetujui dan diselesaikan dalam beberapa menit.

Tabel berikut mencantumkan sumber daya yang area peningkatan kuota dapat disetujui secara otomatis.

Sumber Daya	Kuota default	Kuota maksimum
Kebijakan yang dikelola pelanggan dalam akun	1500	5000
Grup kunci per akun	300	500
Profil instans per akun	1000	5000
Kebijakan terkelola per peran	10	20
Kebijakan terkelola per pengguna	10	20
Panjang kebijakan kepercaya an peran	2048 karakter	4096 karakter
Tabel Per Akun	1000	5000
Sertifikat server per akun	20	1000

IAMKuota Access Analyzer

Untuk daftar titik akhir layanan IAM Access Analyzer dan kuota layanan, lihat titik akhir dan kuota IAMAccess Analyzer di. Referensi Umum AWS

IAMKuota Peran Di Mana Saja

Untuk daftar titik akhir layanan dan kuota layanan IAM Roles Anywhere, lihat Titik akhir dan kuota AWS Identity and Access Management Roles Anywhere di. Referensi Umum AWS

STSpermintaan kuota

AWS STS Layanan ini memiliki kuota permintaan default 600 permintaan per detik per akun, per wilayah. Kuota ini dibagikan di seluruh STS permintaan berikut yang dibuat menggunakan AWS kredensional:

- AssumeRole
- DecodeAuthorizationMessage
- GetAccessKeyInfo
- GetCallerIdentity
- GetFederationToken
- GetSessionToken

Misalnya, jika sebuah Akun AWS membuat 100 GetCallerIdentity permintaan per detik dan 100 AssumeRole panggilan per detik di wilayah yang sama, akun tersebut menghabiskan 200 dari 600 STS permintaan per detik yang tersedia untuk wilayah tersebut.

Untuk AssumeRole permintaan lintas akun, hanya akun yang membuat AssumeRole permintaan yang memengaruhi STS kuota. Akun target tidak memiliki kuota yang dikonsumsi.

> **Note**
>
> Permintaan ke AWS STS oleh prinsipal AWS layanan, seperti yang digunakan untuk mengambil peran untuk digunakan dengan AWS layanan, tidak menggunakan kuota STS permintaan per detik di akun Anda.

Untuk meminta kenaikan STS permintaan kuota, silakan buka tiket dengan AWS dukungan.

IAMdan batas STS karakter

Berikut ini adalah jumlah karakter maksimum dan batas ukuran untuk IAM dan AWS STS. Anda tidak dapat meminta kenaikan untuk batasan berikut.

Deskripsi	Kuota
Alias untuk ID Akun AWS	3–63 karakter
Untuk kebijakan inline	Anda dapat menambahkan kebijakan sebaris sebanyak yang Anda inginkan ke IAM pengguna, peran, atau grup. Tetapi ukuran total kebijakan agregat (ukuran jumlah semua kebijakan sebaris) per entitas tidak dapat melebihi batas berikut: • Ukuran kebijakan pengguna tidak boleh melebihi 2.048 karakter. • Ukuran kebijakan peran tidak dapat melebihi 10.240 karakter. • Ukuran kebijakan grup tidak boleh melebihi 5.120 karakter. **Note** IAMtidak menghitung ruang putih saat menghitung ukuran kebijakan terhadap batas-batas ini.
Untuk kebijakan terkelola	• Ukuran setiap kebijakan terkelola tidak boleh melebihi 6.144 karakter.

Deskripsi	Kuota
	Note IAMtidak menghitung spasi putih saat menghitung ukuran kebijakan terhadap batas ini.
Nama grup	128 karakter
Nama profil instans	128 karakter
Kata sandi untuk profil masuk	1–128 karakter
Jalur	512 karakter
Nama kebijakan	128 karakter
Nama peran	64 karakter **Important** Jika Anda ingin menggunakan peran dengan fitur Switch Role di AWS Management Console, maka gabungan `Path` dan tidak `RoleName` dapat melebihi 64 karakter.

Deskripsi	Kuota
Durasi sesi peran	12 jam Saat Anda mengambil peran dari AWS CLI atauAPI, Anda dapat menggunakan `duration-seconds` CLI parameter atau `DurationSeconds` API parameter untuk meminta sesi peran yang lebih lama. Anda dapat menentukan nilai dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut, yang dapat berkisar antara 1–12 jam. Jika Anda tidak menentukan nilai untuk parameter `DurationSeconds` tersebut, kredensial keamanan Anda berlaku selama satu jam. IAMpengguna yang beralih peran di konsol diberikan durasi sesi maksimum, atau sisa waktu dalam sesi pengguna, mana yang kurang. Pengaturan durasi sesi maksimum tidak membatasi sesi yang diasumsikan oleh AWS layanan. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.
Nama sesi peran	64 karakter

Deskripsi	Kuota
Peran kebijakan sesi	• Ukuran dokumen JSON kebijakan yang diteruskan dan semua ARN karakter kebijakan terkelola yang diteruskan digabungkan tidak dapat melebihi 2.048 karakter. • Anda dapat meneruskan maksimal 10 kebijakan terkelola ARNs saat membuat sesi. • Anda hanya dapat meneruskan satu dokumen JSON kebijakan saat membuat sesi sementara untuk peran atau pengguna gabungan secara terprogram. • Selain itu, AWS konversi memampatkan kebijakan sesi dan tag sesi yang diteruska n ke dalam format biner yang dikemas yang memiliki batas terpisah. Elemen respons `PackedPolicySize` menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas. • Kami menyarankan Anda untuk lulus kebijakan sesi menggunakan AWS CLI atau AWS API. AWS Management Console Mungkin menambahkan informasi sesi konsol tambahan ke kebijakan yang dikemas.

Deskripsi	Kuota
Peran tanda sesi	• Tag sesi harus memenuhi batas kunci tag 128 karakter dan batas nilai tag 256 karakter. • Anda dapat meneruskan hingga 50 tanda sesi. • AWS Konversi memampatkan kebijakan sesi dan tag sesi yang diteruskan ke dalam format biner yang dikemas yang memiliki batas terpisah. Anda dapat meneruskan tag sesi menggunakan AWS CLI atau AWS API. Elemen respons `PackedPolicySize` menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas.
SAMLotentikasi respon base64 dikodekan	100.000 karakter Batas karakter ini berlaku untuk `assume-role-with-saml` CLIatau `AssumeRoleWithSAML` APIoperasi.
Tombol tanda	128 karakter Batas karakter ini berlaku untuk tag pada IAM sumber daya dan tag sesi.
Nilai tanda	256 karakter Batas karakter ini berlaku untuk tag pada IAM sumber daya dan tag sesi. Nilai tag bisa kosong yang berarti nilai tag dapat memiliki panjang 0 karakter.

Deskripsi	Kuota
Unik IDs dibuat oleh IAM	128 karakter. Sebagai contoh: • Pengguna IDs yang dimulai dengan `AIDA` • Grup IDs yang dimulai dengan `AGPA` • Peran IDs yang dimulai dengan `AROA` • Kebijakan terkelola IDs yang dimulai dengan `ANPA` • Sertifikat server IDs yang dimulai dengan `ASCA` **Note** Ini tidak dimaksudkan untuk menjadi daftar lengkap, juga bukan jaminan bahwa IDs jenis tertentu dimulai hanya dengan kombinasi huruf yang ditentukan.
Nama pengguna	64 karakter

Titik VPC akhir antarmuka

Jika Anda menggunakan Amazon Virtual Private Cloud (AmazonVPC) untuk meng-host AWS sumber daya, Anda dapat membuat koneksi pribadi antara Anda VPC dan AWS Identity and Access Management (IAM) atau AWS Security Token Service (AWS STS). Anda dapat menggunakan koneksi ini untuk mengaktifkan IAM atau AWS STS untuk berkomunikasi dengan sumber daya Anda di Anda VPC tanpa melalui internet publik.

Amazon VPC adalah AWS layanan yang dapat Anda gunakan untuk meluncurkan AWS sumber daya dalam jaringan virtual yang Anda tentukan. Dengan aVPC, Anda memiliki kontrol atas pengaturan jaringan Anda, seperti rentang alamat IP, subnet, tabel rute, dan gateway jaringan. Untuk menghubungkan Anda VPC ke IAM atau AWS STS, Anda menentukan VPCtitik akhir antarmuka untuk setiap layanan. Titik akhir menyediakan konektivitas yang andal dan dapat diskalakan ke atau

IAM AWS STS tanpa memerlukan gateway internet, instance terjemahan alamat jaringan (NAT), atau VPN koneksi. Untuk informasi selengkapnya, lihat Apa itu AmazonVPC? di Panduan VPC Pengguna Amazon.

VPCEndpoint antarmuka didukung oleh AWS PrivateLink sebuah AWS teknologi yang memungkinkan komunikasi pribadi antara AWS layanan menggunakan elastic network interface dengan alamat IP pribadi. Untuk informasi selengkapnya, silakan lihat AWS PrivateLink untuk AWS Layanan.

Informasi berikut adalah untuk pengguna AmazonVPC. Untuk informasi selengkapnya, lihat Memulai Amazon VPC di Panduan VPC Pengguna Amazon.

Topik

- VPCketersediaan titik akhir
- Buat VPC titik akhir untuk IAM
- Buat VPC titik akhir untuk AWS STS

VPCketersediaan titik akhir

> **Important**
>
> VPCTitik akhir antarmuka untuk hanya IAM dapat dibuat di Wilayah tempat bidang IAM kontrol berada. Jika Anda VPC berada di Wilayah yang berbeda dari Wilayah pesawat IAM kontrol, Anda harus menggunakan AWS Transit Gateway untuk memungkinkan akses ke VPC titik akhir IAM antarmuka dari Wilayah lain. Untuk informasi selengkapnya, lihat Buat VPC titik akhir untuk IAM.

IAMsaat ini mendukung VPC titik akhir di Wilayah berikut:

- AS Timur (Virginia Utara)
- Tiongkok (Beijing)

AWS STS saat ini mendukung VPC titik akhir di Wilayah berikut:

- AS Timur (N. Virginia)
- AS Timur (Ohio)

- AS Barat (California Utara)
- AS Barat (Oregon)
- Afrika (Cape Town)
- Asia Pasifik (Hong Kong)
- Asia Pasifik (Hyderabad)
- Asia Pasifik (Jakarta)
- Asia Pasifik (Melbourne)
- Asia Pasifik (Mumbai)
- Asia Pasifik (Osaka)
- Asia Pasifik (Seoul)
- Asia Pasifik (Singapura)
- Asia Pasifik (Sydney)
- Asia Pasifik (Tokyo)
- (Canada (Central)
- Kanada Barat (Calgary)
- China (Beijing)
- Tiongkok (Ningxia)
- Eropa (Frankfurt)
- Eropa (Irlandia)
- Eropa (London)
- Eropa (Milan)
- Eropa (Paris)
- Eropa (Spanyol)
- Eropa (Stockholm)
- Eropa (Zürich)
- Israel (Tel Aviv)
- Timur Tengah (Bahrain)
- Timur Tengah (UAE)
- Amerika Selatan (Sao Paulo)
- AWS GovCloud (AS-Timur)

- AWS GovCloud (AS-Barat)

Buat VPC titik akhir untuk IAM

Untuk mulai menggunakan IAM dengan AndaVPC, buat VPC titik akhir antarmuka untukIAM. Untuk informasi selengkapnya, lihat Mengakses AWS layanan menggunakan VPC titik akhir antarmuka di Panduan VPC Pengguna Amazon.

VPCTitik akhir antarmuka untuk hanya IAM dapat dibuat di Wilayah tempat bidang IAM kontrol berada. Secara komersial Wilayah AWS, pesawat IAM kontrol terletak di Wilayah AS Timur (Virginia N.) (us-east-1). Nama layanan VPC endpoint AWS PrivateLink antarmuka untuk IAM adalah`com.amazonaws.iam`. Untuk daftar VPC titik akhir dukungan Wilayah AWS tersebutIAM, lihatVPCketersediaan titik akhir.

Jika Anda VPC berada di Wilayah yang berbeda dari Wilayah bidang IAM kontrol, Anda harus menggunakan AWS Transit Gateway untuk mengizinkan akses ke VPC titik akhir IAM antarmuka dari Wilayah lain.

Untuk mengakses VPC titik akhir IAM antarmuka dari VPC di Wilayah yang berbeda menggunakan AWS Transit Gateway

1. Buat gateway transit, atau gunakan gateway transit yang ada untuk menghubungkan cloud pribadi virtual Anda (VPCs). Gateway transit diperlukan untuk setiap Wilayah. Untuk informasi selengkapnya, lihat Membuat gateway transit di AWS Transit Gateway Panduan.
2. Buat VPC lampiran gateway transit untuk menghubungkan masing-masing VPC ke gateway transit. Untuk informasi selengkapnya, lihat Membuat lampiran gateway transit ke VPC dalam AWS Transit Gateway Panduan.
3. Buat lampiran VPC peering gateway transit untuk merutekan lalu lintas antara VPCs peered. Untuk informasi selengkapnya, lihat Membuat lampiran peering di AWS Transit Gateway Panduan.

> **Note**
>
> VPCKoneksi peering juga dapat merutekan lalu lintas antara peeredVPCs, tetapi metode ini tidak berskala baik dengan sejumlah besar. VPCs Alih-alih VPC mengintip, kami merekomendasikan AWS Transit Gateway peering lampiran yang meningkatkan VPC dan manajemen jaringan lokal melalui hub pusat yang dapat diskalakan. Untuk informasi

selengkapnya tentang koneksi VPC peering, lihat Bekerja dengan koneksi VPC peering di Panduan VPCPeering Amazon.

Buat VPC titik akhir untuk AWS STS

Untuk mulai menggunakan AWS STS dengan AndaVPC, buat VPC titik akhir antarmuka untuk AWS STS. Untuk informasi selengkapnya, lihat Mengakses AWS layanan menggunakan VPC titik akhir antarmuka di Panduan VPC Pengguna Amazon.

Setelah Anda membuat VPC endpoint, Anda harus menggunakan endpoint regional yang cocok untuk mengirim permintaan Anda AWS STS . AWS STS merekomendasikan agar Anda menggunakan kedua `setEndpoint` metode `setRegion` dan untuk melakukan panggilan ke titik akhir Regional. Anda hanya dapat menggunakan metode `setRegion` untuk Wilayah yang diaktifkan secara manual, seperti Asia Pacific (Hong Kong). Dalam hal ini, panggilan diarahkan ke titik akhir STS Regional. Untuk mempelajari cara mengaktifkan sebuah Wilayah secara manual, lihat Mengelola Wilayah AWS dalam Referensi Umum AWS. Jika Anda hanya menggunakan metode `setRegion` untuk Wilayah yang diaktifkan secara default, panggilan diarahkan ke titik akhir global `https://sts.amazonaws.com` .

Saat Anda menggunakan endpoint regional, AWS STS memanggil AWS layanan lain menggunakan titik akhir publik atau VPC titik akhir antarmuka pribadi, mana saja yang digunakan. Misalnya, asumsikan bahwa Anda telah membuat VPC titik akhir antarmuka untuk AWS STS dan telah meminta kredensional sementara AWS STS dari sumber daya yang berada di Anda. VPC Dalam hal ini, kredenal ini mulai mengalir melalui VPC titik akhir antarmuka secara default. Untuk informasi selengkapnya tentang membuat permintaan Regional menggunakan AWS STS, lihatKelola AWS STS dalam sebuah Wilayah AWS.

AWS layanan yang bekerja dengan IAM

AWS Layanan yang tercantum di bawah ini dikelompokkan menurut abjad dan mencakup informasi tentang IAM fitur apa yang mereka dukung:

- Layanan — Anda dapat memilih nama layanan untuk melihat AWS dokumentasi tentang IAM otorisasi dan akses untuk layanan tersebut.
- Tindakan – Anda dapat menentukan tindakan individu dalam kebijakan. Jika layanan tidak mendukung fitur ini, maka Semua tindakan dipilih di editor visual. Dalam dokumen JSON kebijakan,

Anda harus menggunakan * `Action` elemen. Untuk daftar tindakan di setiap layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

- Izin tingkat sumber daya — Anda dapat menggunakan ARNsuntuk menentukan sumber daya individual dalam kebijakan. Jika layanan tidak mendukung fitur ini, maka Semua sumber daya dipilih di editor visual kebijakan. Dalam dokumen JSON kebijakan, Anda harus menggunakan * `Resource` elemen. Beberapa tindakan, seperti `List*` tindakan, tidak mendukung penentuan ARN karena dirancang untuk mengembalikan banyak sumber daya. Jika suatu layanan mendukung fitur ini untuk beberapa sumber daya tetapi tidak yang lain, itu ditunjukkan oleh Partial dalam tabel. Lihat dokumentasi untuk layanan tersebut untuk informasi selengkapnya.
- Kebijakan berbasis sumber daya – Anda dapat melampirkan kebijakan berbasis sumber daya pada sumber daya dalam layanan. Kebijakan berbasis sumber daya mencakup `Principal` elemen untuk menentukan IAM identitas mana yang dapat mengakses sumber daya tersebut. Untuk informasi selengkapnya, lihat Kebijakan berbasis identitas dan kebijakan berbasis sumber daya.
- ABAC(otorisasi berdasarkan tag) — Untuk mengontrol akses berdasarkan tag, Anda memberikan informasi tag dalam elemen kondisi kebijakan menggunakan`aws:ResourceTag/`*`key-name`*`,aws:RequestTag/`*`key-name`*, atau kunci `aws:TagKeys` kondisi. Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial. Untuk informasi selengkapnya tentang mendefinisikan izin berdasarkan atribut seperti tanda, lihat Tentukan izin berdasarkan atribut dengan otorisasi ABAC. Untuk melihat tutorial dengan langkah-langkah pengaturanABAC, lihat Menggunakan kontrol akses berbasis atribut () ABAC.
- Kredensi sementara — Anda dapat menggunakan kredenal jangka pendek yang diperoleh saat masuk menggunakan Pusat IAM Identitas, beralih peran di konsol, atau yang Anda hasilkan menggunakan AWS STS di atau. AWS CLI AWS API Anda dapat mengakses layanan dengan nilai Tidak hanya saat menggunakan kredensil IAM pengguna jangka panjang Anda. Ini termasuk nama pengguna dan kata sandi atau access key pengguna. Untuk informasi selengkapnya, lihat Kredensi keamanan sementara di IAM.
- Peran yang ditautkan dengan layanan – Sebuah peran yang berkaitan dengan layanan adalah jenis peran layanan khusus yang memberikan izin layanan untuk mengakses sumber daya pada layanan lain atas nama Anda. Pilih tautan Ya atau Sebagian untuk melihat dokumentasi layanan yang mendukung peran ini. Kolom ini tidak menunjukkan apakah layanan menggunakan peran layanan standar. Untuk informasi selengkapnya, lihat Peran terkait layanan.
- Informasi selengkapnya – Jika layanan tidak sepenuhnya mendukung fitur, Anda dapat meninjau catatan kaki untuk entri untuk melihat pembatasan dan tautan ke informasi terkait.

Layanan yang bekerja dengan IAM

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Account Management	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Activate Console	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Amplify Admin	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Amplify	Ya	Ya	T[illegible]	Sebag	Ya	Tidak
AWS Amplify Pembuat UI	Ya	Ya	Tida	Ya	Ya	Tidak
Apache Kafka untuk cluster APIs Amazon MSK	Ya	Ya	Tida	Tida	Ya	Tidak
APIGerbang Amazon	Ya	Ya	Ya	Tida	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Manajemen Amazon API Gateway	Ya	Ya	Tida	Ya	Ya	Tidak
Manajemen API Gateway Amazon V2	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Studio Aplikasi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS App2Container	Ya	Tida	Tida	Tida	Ya	Tidak
AWS AppConfig	Ya	Ya	Tida	Ya	Ya	Tidak
AWS AppFabric	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon AppFlow	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon AppIntegrations	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Penskalaan Otomatis Aplikasi	Ya	Ya	Tida	Ya	Ya	Ya
AWS Profiler Biaya Aplikasi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Aplikasi Discovery Arsenal	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Application Discovery Service	Ya	Tida	Tida	Tida	Ya	Ya
AWS Application Migration Service	Ya	Ya	Tida	Ya	Ya	Ya
AWS Layanan Transformasi Aplikasi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS App Mesh	Ya	Ya	Tida	Ya	Ya	Ya
AWS App Mesh Pratinjau	Ya	Ya	Tida	Tida	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Pelari Aplikasi	Ya	Ya	Tida	Ya	Ya	Ya
Amazon AppStream 2.0	Ya	Ya	Tida	Ya	Ya	Tidak
AWS AppSync	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Artifact	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Athena	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Audit Manager	Ya	Ya	Tida	Ya	Ya	Ya
AWS Auto Scaling	Ya	Tida	Tida	Tida	Ya	Ya
AWS Pertukaran Data B2B	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Backup	Ya	Ya	Ya	Ya	Ya	Ya
AWS Backup Gerbang	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Backup penyimpanan	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Batch	Sebag	Tida	Ya	Ya	Ya	Ya
Amazon Bedrock	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Billing and Cost Management	Ya	Tida	Tida	Tida	Ya	Ya
AWS Billing and Cost Management Ekspor Data	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Billing Conductor	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon Braket	Ya	Ya	Tida	Ya	Ya	Ya
AWS Layanan Anggaran	Ya	Ya	Tida	Tida	Tida	Tidak
AWS BugBust	Ya	Ya	Tida	Ya	Ya	Ya
AWS Certificate Manager (ACM)	Ya	Ya	Tida	Ya	Ya	Ya
AWS Chatbot	Ya	Ya	Tida	Tida	Ya	Ya
Amazon Chime	Ya	Ya	Tida	Ya	Ya	Ya
AWS Clean Rooms	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Clean Rooms ML	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Client VPN	Ya	Ya	Tida	Tida	Ya	Ya
AWS Cloud9	Ya	Ya	Ya	Ya	Ya	Ya
AWS Cloud Kontrol API	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Cloud Directory	Ya	Ya	Tida	Tida	Ya	Tidak
AWS CloudFormation	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon CloudFront	Ya	Ya	Tida	Sebag	Ya	Sebagian (Info)
Amazon CloudFront KeyValueStore	Ya	Ya	Tida	Tida	Ya	Tidak
AWS CloudHSM	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Cloud Map	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon CloudSearch	Ya	Ya	Tida	Tida	Ya	Tidak
AWS CloudShell	Ya	Ya	Tida	Tida	Ya	Tidak
AWS CloudTrail	Ya	Ya	Sebag (Info)	Ya	Ya	Ya
AWS CloudTrail Data	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon CloudWatch	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)
Wawasan CloudWatch Aplikasi Amazon	Ya	Tida	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Sinyal CloudWatch Aplikasi Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon CloudWatch Terbukti	Ya	Ya	Tida	Ya	Ya	Tidak
Monitor CloudWatch Internet Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
CloudWatch Log Amazon	Ya	Ya		Sebag	Ya	Ya
Monitor CloudWatch Jaringan Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Manajer Akses CloudWatch Observabilitas Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon CloudWatch RUM	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon CloudWatch Synthetics	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS CodeArtifact	Ya	Ya	Ya	Ya	Ya	Tidak
AWS CodeBuild	Ya	Ya	(Info)	Sebag (Info)	Ya	Tidak
Amazon CodeCatalyst	Ya	Ya	Tida	Ya	Ya	Ya
AWS CodeCommit	Ya	Ya	Tida	Ya	Ya	Tidak
AWS CodeConnections	Ya	Ya	Tida	Ya	Ya	Tidak
AWS CodeDeploy	Ya	Ya	Tida	Ya	Ya	Tidak
AWS CodeDeploy layanan perintah host aman	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon CodeGuru Profiler	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
CodeGuru Peninjau Amazon	Ya	Ya	Tida	Ya	Ya	Ya
CodeGuru Keamanan Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
AWS CodePipeline	Sebag	Tida	Ya	Ya	Tidak	
AWS CodeStar	Sebag	Tida	Ya	Ya	Tidak	
AWS CodeStar Koneksi	Ya	Ya	Tida	Ya	Ya	Ya
AWS CodeStar Pemberitahuan	Ya	Ya	Tida	Ya	Ya	Ya
Amazon CodeWhisperer	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Cognito	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Sinkronisasi Amazon Cognito	Ya	Ya	Tida	Tida	Ya	Ya
Kumpulan pengguna Amazon Cognito	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Comprehend	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Comprehend Medical	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Compute Optimizer	Ya	Tida	Tida	Tida	Ya	Ya
AWS Config	Ya	Seba (Info)	Tida	Ya	Ya	Ya
Amazon Connect	Ya	Ya	Tida	Ya	Ya	Ya
Kasus Amazon Connect	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Profil Pelanggan Amazon Connect	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Connect Komunikasi keluar volume tinggi	Ya	Ya	Tida	Ya	Ya	Tidak
ID Suara Amazon Connect	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Console Mobile Application	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Penagihan Konsolidasi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Katalog Kontrol	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Control Tower	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Cost and Usage Report	Ya	Ya	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Cost Explorer	Ya	Ya	Tida	Ya	Ya	Tidak
Hub Optimisasi Biaya AWS	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Layanan Verifikasi Pelanggan	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Database Migration Service	Ya	Ya	Tida (Info)	Ya	Ya	Ya
Layanan Metadata Kueri Basis Data	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Data Exchange	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Data Lifecycle Manager	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Data Pipeline	Ya	Ya	Tida	Sebag	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS DataSync	 Ya	 Ya	 Tida	 Ya	 Ya	 Ya
Amazon DataZone	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Batas Waktu Cloud	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS DeepComposer	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS DeepRacer	 Ya	 Ya	 Tida	 Ya	 Ya	 Ya
Amazon Detective	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Device Farm	 Ya	 Ya	 Tida	 Ya	Ya	Ya
DevOpsGuru Amazon	Ya	Ya	Tida	Tida	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Alat diagnostik	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Direct Connect	Ya	Ya	Tida	Ya	Ya	Ya
AWS Directory Service	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Directory Service Data	Ya	Ya	Tida	Ya	Ya	Tidak
Cluster Elastis Amazon DocumentDB	Ya	Ya	Tida	Ya	Ya	Ya
Akselerator Amazon DynamoDB () DAX	Ya	Ya	Tida	Tida	Ya	Ya
Amazon DynamoDB	Ya	Ya	Ya	Tida	Ya	Tidak
Amazon Elastic Compute Cloud (AmazonEC2)	Ya	Sebag	Tida	Ya	Ya	Sebagian (Info)

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
EC2Auto Scaling Amazon	Ya	Ya	Tida	Ya	Ya	Ya
EC2Image Builder	Ya	Ya	Tida	Ya	Ya	Ya
Amazon EC2 Instans Connect	Ya	Ya	Tida	Ya	Ya	Ya
Amazon ElastiCache	Ya	Ya	Tida	Ya	Ya	Ya
AWS Elastic Beanstalk	Sebag	Tida	Ya	Ya	Ya	Ya
Toko Blok Elastis Amazon (AmazonEBS)	Sebag	Tida	Ya	Ya	Tidak	
Registri Wadah Elastis Amazon (AmazonECR)	Ya	Ya	Ya	Ya	Ya	Ya
Registri Kontainer Elastis Amazon Publik (Amazon ECR Publik)	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Layanan Kontainer Elastis Amazon (AmazonECS)	Ya	Sebag (Info)	Tida	Ya	Ya	Ya
AWS Elastic Disaster Recovery	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Elastic File System (AmazonEFS)	Ya	Ya	Ya	Sebag	Ya	Ya
Amazon Elastic Inference	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Elastic Kubernetes Service (Amazon) EKS	Ya	Ya	Tida	Ya	Ya	Ya
Auth Amazon Elastic Kubernetes Service (Amazon) EKS	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Elastic Load Balancing	Ya	Sebag	Tida	Sebag	Ya	Ya
Amazon Elastic Transcoder	Ya	Ya	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Peralatan Elemental dan Layanan Aktivasi Perangkat Lunak	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Peralatan dan Perangkat Lunak Elemental	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Elemental MediaConnect	Ya	Ya	Tida	Tida	Ya	Ya
AWS Elemental MediaConvert	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Elemental MediaLive	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Elemental MediaPackage	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)
AWS Elemental MediaPackage V2	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Elemental MediaPackage VOD	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)
AWS Elemental MediaStore	Ya	Ya	Ya	Ya	Ya	Tidak
AWS Elemental MediaTailor	Ya	Ya	Tida	Ya	Ya	Ya
AWS Kasus Dukungan Elemental	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Konten Dukungan Elemental	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon EMR	Ya	Ya	Tida	Ya	Ya	Ya
Amazon EMR di EKS	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Tanpa EMR Server	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Pesan Pengguna Akhir Sosial	Ya	Ya	Tida	Ya	Ya	Ya
AWS Resolusi Entitas	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon EventBridge	Ya	Ya	Ya	Ya	Ya	Tidak
EventBridge Pipa Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
EventBridge Penjadwal Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
EventBridge Skema Amazon	Ya	Ya	Ya	Ya	Ya	Tidak
AWS Layanan Injeksi Kesalahan	Ya	Ya	Tida	Ya	Ya	Ya
Amazon FinSpace	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon FinSpace API	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Firewall Manager	Ya	Ya	Tida	Ya	Y	Sebagian
Fleet Hub for AWS IoT Device Management	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Forecast	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Fraud Detector	Ya	Ya	Tida	Ya	Ya	Tidak
Gratis RTOS	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Tingkat Gratis	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon FSx	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon GameLift	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Global Accelerator	Ya	Ya	Tida	Ya	Ya	Ya
AWS Glue	Ya	Ya	 Sebag		Ya	Tidak
AWS Glue DataBrew	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Ground Station	Ya	Ya	Tida	Ya	Ya	Ya
Pelabelan Ground Truth Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon GuardDuty	Ya	Ya	Tida	Ya	Ya	Ya
AWS Health APIsDan Pemberitahuan	Ya	Ya	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS HealthImaging	Ya	Ya	Tida	Ya	Ya	Tidak
AWS HealthLake	Ya	Ya	Tida	Ya	Ya	Tidak
AWS HealthOmics	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IAM Identity Center	Ya	Ya	T...	Sebag	Ya	Ya
IAMDirektori Pusat Identitas	Ya	Tida	Tida	Tida	Ya	Tidak
IAMToko Identitas Pusat Identitas	Ya	Ya	Tida	Tida	Ya	Tidak
IAMOIDCLayanan Pusat Identitas	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Identity and Access Management (IAM)	Ya	Ya	Se... (Info)	Se... (Info)	Sebag (Info)	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Identity and Access Management Penganalisis Akses	Ya	Ya	Tida	Ya	Ya	Sebagian
AWS Identity and Access Management Peran Di Mana Saja	Ya	Ya	Tida	Ya	Ya	Ya
AWS Auth Toko Identitas	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Sinkronisasi Identitas	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Import/Export	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Inspector	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Inspector Klasik	Ya	Tida	Tida	Tida	Ya	Ya
Amazon InspectorScan	Ya	Tida	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Layanan Video Interaktif Amazon	Ya	Ya	Tida	Ya	Ya	Ya
Obrolan Layanan Video Interaktif Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Invoicing	Ya	Tida	Tida	Tida	Ya	Tidak
AWS IoT 1-Click	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IoT Analytics	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IoT	Ya	Ya	Sebag (Info)	Ya	Ya	Tidak
AWS IoT Core Penasihat Perangkat	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IoT Penguji Perangkat	Ya	Tida	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS IoT Events	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IoT FleetWise	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IoT Greengrass	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IoT Greengrass V2	Ya	Ya	Tida	Sebag	Ya	Tidak
AWS IoT Lowongan DataPlane	Ya	Ya	Tida	Tida	Ya	Tidak
AWS IoT SiteWise	Ya	Ya	Tida	Ya	Ya	Ya
AWS IoT TwinMaker	Ya	Ya	Tida	Ya	Ya	Ya
AWS IoT Wireless	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS IQ	Ya	Ya	Tida	Tida	Ya	Ya
AWS Izin IQ	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Kendra	Ya	Ya	Tida	Ya	Ya	Tidak
Peringkat Cerdas Amazon Kendra	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Key Management Service (AWS KMS)	Ya	Ya	Ya	Ya	Ya	Ya
Amazon Keyspaces (untuk Apache Cassandra)	Ya	Ya	Tida	Ya	Ya	Ya
Layanan Dikelola Amazon untuk Apache Flink	Ya	Ya	Tida	Ya	Ya	Tidak
Layanan Dikelola Amazon untuk Apache Flink V2	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon Data Firehose	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Kinesis Data Streams	Ya	Ya	Ya	Ya	Ya	Tidak
Aliran Video Amazon Kinesis	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Lake Formation	Ya	Tida	Tida	Tida	Ya	Ya
AWS Lambda	Ya	Ya	Ya	Sebag (Info)	Ya	Sebagian (Info)
AWS Launch Wizard	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Lex	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Lex V2	Ya	Ya	Ya	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS License Manager	Ya	Ya	Tida	Ya	Ya	Ya
AWS License Manager Manajer Langganan Linux	Ya	Tida	Tida	Tida	Ya	Tidak
AWS License Manager Langganan Pengguna	Ya	Tida	Tida	Tida	Ya	Ya
Amazon Lightsail	Ya	Seba (Info)	Tida	Sebag (Info)	Ya	Ya
Amazon Location Service	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Lookout for Equipment	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Lookout for Metrics	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Lookout for Vision	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon Machine Learning	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Macie	Ya	Ya	Tida	Ya	Ya	Ya
AWS Mainframe Modernization	Ya	Ya	Tida	Ya	Ya	Ya
AWS Mainframe Modernization Pengujian Aplikasi	Ya	Ya	Tida	Ya	Ya	Tidak
Blockchain yang Dikelola Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Kueri Blockchain yang Dikelola Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
Grafana yang Dikelola Amazon	Ya	Ya	Tida	Ya	Ya	Ya
Layanan Terkelola Amazon untuk Prometheus	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon Managed Streaming for Apache Kafka () MSK	 Ya	 Ya	Sebag (Info)	 Ya	 Ya	 Ya
Amazon Managed Streaming for Kafka Connect	 Ya	 Ya	 Tida	 Tida	 Ya	 Ya
Alur Kerja Terkelola Amazon untuk Apache Airflow	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Marketplace	 Ya	 Tida	 Tida	 Tida	 Ya	 Ya
AWS Marketplace Katalog	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Marketplace Commerce Analytics	 Ya	 Tida	 Tida	 Tida	 Tida	 Tidak
AWS Marketplace Layanan Deployment	 Ya	 Ya	 Tida	 Ya	 Ya	Tidak
AWS Marketplace Penemuan	Ya	Tida	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Marketplace Entitlement Service	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Marketplace Layanan Image Building	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
Portal Manajemen AWS Marketplace	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Marketplace Metering Service	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Marketplace Marketplace Pribadi	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Marketplace Integrasi Sistem Pengadaan	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Marketplace Pelaporan	 Ya	 Ya	 Tida	 Tida	Ya	Tidak
AWS Marketplace Pelaporan Penjual	Ya	Ya	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Marketplace Wawasan Vendor	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Mechanical Turk	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon MediaImport	Ya	Tida	Tida	Tida	Tida	Tidak
Amazon MemoryDB	Ya	Ya	Tida	Ya	Ya	Ya
Layanan Pengiriman Pesan Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
Layanan Gateway Pesan Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Microservice Extractor for .NET	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Kredit Program Percepatan Migrasi	Ya	Ya	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Migration Hub	Ya	Ya	Tida	Tida	Ya	Ya
AWS Migration Hub Orkestrator	Ya	Ya	Tida	Ya	Ya	Ya
AWS Migration Hub Refactor Spaces	Ya	Ya	Ya	Ya	Ya	Ya
AWS Migration Hub Rekomendasi Strategi	Ya	Tida	Tida	Tida	Ya	Ya
Amazon Monitron	Ya	Ya	Tida	Ya	Ya	Ya
Amazon MQ	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Neptune	Ya	Ya	Tida	Tida	Ya	Ya
Analisis Amazon Neptunus	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Network Firewall	Ya	Ya	Tida	Ya	Ya	Ya
AWS Network Manager	Ya	Ya	Tida	Ya	Ya	Ya (Info)
AWS Network Manager Obrolan	Ya	Tida	Tida	Tida	Ya	Tidak
Studio Amazon Nimble	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Satu Perusahaan	Ya	Ya	Tida	Ya	Ya	Tidak
OpenSearchTertelan Amazon	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Tanpa OpenSearch Server	Ya	Ya	Tida	Ya	Ya	Ya
OpenSearch Layanan Amazon	Ya	Ya	Ya	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS OpsWorks	Ya	Ya	Tida	Tida	Ya	Tidak
AWS OpsWorks Manajemen Konfigurasi	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Organizations	Ya	Ya	Ya	Ya	Tida	Ya
AWS Outposts	Ya	Ya	Tida	Ya	Ya	Ya
AWS Panorama	Ya	Ya	Tida	Ya	Ya	Ya
AWS Layanan Komputasi Paralel	Ya	Ya	Tida	Ya	Ya	Ya
AWS Partner Manajemen akun pusat	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Payment Cryptography	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Pembayaran	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Wawasan Performa	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Personalisasi	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Pinpoint	Ya	Ya	Tida	Ya	Ya	Tidak
Layanan Email Amazon Pinpoint	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Pinpoint SMS dan Layanan Suara	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Pinpoint SMS dan Layanan Suara V2	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Polly	Ya	Ya	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Daftar Harga AWS	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS 5G pribadi	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Private CA Konektor untuk Active Directory	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Private CA Konektor untuk SCEP	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Private Certificate Authority (AWS Private CA)	 Ya	 Ya	 Ya	 Ya	 Ya	 Tidak
AWS Proton	 Ya	 Ya	 Tida	 Ya	 Ya	 Ya
AWS Konsol Pesanan Pembelian	 Ya	 Ya	 Tida	 Ya	Ya	Tidak
Amazon Q Bisnis	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Aplikasi Q Bisnis Amazon Q	Ya	Ya	Tida	Tida	Ya	Tidak
Pengembang Amazon Q	Ya	Tida	Tida	Tida	Ya	Ya
Amazon Q di Connect	Ya	Ya	Tida	Ya	Ya	Tidak
Database Buku Besar Amazon Quantum (AmazonQLDB)	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon QuickSight	Ya	Ya	Tida	Ya	Ya	Tidak
RDSData Amazon API	Ya	Ya	Tida	Tida	Ya	Tidak
RDSIAMOtentikasi Amazon	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Tempat Sampah Daur Ulang	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon Redshift	Ya	Ya	Tida	Ya	Ya	Ya
Data Pergeseran Merah Amazon API	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Redshift Tanpa Server	Ya	Ya	Ya	Ya	Ya	Tidak
Amazon Rekognition	Ya	Ya	Sebag (Info)	Ya	Ya	Tidak
Layanan Database Relasional Amazon (RDSAmazon) (Info)	Ya	Ya	Tida	Ya	Ya	Ya
AWS re:Post Pribadi	Ya	Ya	Tida	Ya	Ya	Ya
AWS Resilience Hub	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Resource Access Manager (AWS RAM)	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Penjelajah Sumber Daya AWS	Ya	Ya	Tida	Ya	Ya	Ya
AWS Resource Groups	Ya	Ya	Tida	Ya	Sebag (Info)	Ya
AWS Resource Groups Tagging API	Ya	Tida	Tida	Tida	Ya	Tidak
Portal RHEL Basis Pengetahuan Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
AWS RoboMaker	Ya	Ya	Tida	Ya	Ya	Ya
Rute Amazon 53	Ya	Ya	Tida	Tida	Ya	Tidak
Pengontrol Pemulihan Aplikasi Amazon Route 53 - Zonal Shift	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Route 53 Domain	Ya	Tida	Tida	Tida	Tida	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Profil Amazon Route 53	Ya	Ya	Tida	Ya	Ya	Tidak
Cluster Pemulihan Amazon Route 53	Ya	Ya	Tida	Tida	Ya	Tidak
Config Kontrol Pemulihan Amazon Route 53	Ya	Ya	Tida	Ya	Ya	Tidak
Kesiapan Pemulihan Amazon Route 53	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Route 53 Resolver	Ya	Ya	Tida	Ya	Ya	Ya
Amazon S3 Ekspres	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon S3 Glacier	Ya	Ya	Ya	Ya	Ya	Sebagian
Amazon SageMaker	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Kemampuan SageMaker geospasial Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon SageMaker Ground Truth Sintetis	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon SageMaker dengan MLflow	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Savings Plans	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Secrets Manager	Ya	Ya	Ya	Ya	Ya	Tidak
AWS Security Hub	Ya	Ya	Tida	Ya	Ya	Ya
Danau Keamanan Amazon	Ya	Ya	Tida	Tida	Ya	Ya
AWS Security Token Service (AWS STS)	Ya	Seba (Info)	Tida	Ya	Sebag (Info)	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Serverless Application Repository	Ya	Ya	Ya	Tida	Ya	Tidak
AWS Service Catalog	Ya	Ya	Tida	Ya	Ya	Ya
Service Quotas	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Shield	Ya	Ya	Tida	Ya	Ya	Ya
AWS Signer	Ya	Ya	Ya	Ya	Ya	Tidak
AWS Masuk	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon SimpleDB	Ya	Ya	Tida	Tida	Ya	Tidak
Layanan Email Sederhana Amazon - Manajer Surat	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Layanan Email Sederhana Amazon (AmazonSES) v2	Ya	Sebag (Info)	Ya	Ya	Sebag (Info)	Ya
Layanan Pemberitahuan Sederhana Amazon (AmazonSNS)	Ya	Ya	Ya	Ya	Ya	Tidak
Layanan Antrian Sederhana Amazon (AmazonSQS)	Ya	Ya	Ya	Sebag	Ya	Tidak
Amazon Simple Storage Service (Amazon S3)	Ya	Ya	Ya	Sebag (Info)	Ya	Sebagian (Info)
Layanan Penyimpanan Sederhana Amazon (Amazon S3) Objek Lambda	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Simple Storage Service (Amazon S3) aktif AWS Outposts	Ya	Ya	Ya	Tida	Ya	Ya
Layanan Alur Kerja Sederhana Amazon (AmazonSWF)	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS SimSpacePenenun	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Site-to-Site VPN	Ya	Ya	Tida	Tida	Ya	Ya
AWS Snowball	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Snowball Tepi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Snow Device Management	Ya	Ya	Tida	Ya	Ya	Tidak
AWS SQL Workbench	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Step Functions	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Storage Gateway	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Rantai Pasokan AWS	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Support App in Slack	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Support	Ya	Tida	Tida	Tida	Ya	Ya
AWS Support Rencana	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Support Rekomendasi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Keberlanjutan	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Systems Manager	Ya	Ya	Sebag	Ya	Ya	Ya
AWS Systems Manager untuk SAP	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Systems Manager GUIConnect	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Systems Manager Incident Manager	Ya	Ya	Ya	Ya	Ya	Ya
AWS Systems Manager Incident Manager Kontak	Ya	Ya	Ya	Tida	Ya	Tidak
AWS Systems Manager Pengaturan Cepat	Ya	Ya	Tida	Ya	Ya	Tidak
Penyunting Tag	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Pengaturan Pajak	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Pembangun Jaringan Telekomunikasi	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Texttract	Ya	Tida	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon Timestream	Ya	Ya	Tida	Ya	Ya	Tidak
Influxdb Amazon Timestream	Ya	Ya	Tida	Ya	Ya	Ya
AWS Tiros API (untuk Reachability Analyzer)	Ya	Tida	Tida	Tida	Tida	Tidak
Amazon Transcribe	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Transfer Family	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Translate	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Trusted Advisor	Sebag (Info)	Ya	Tida	Tida	Sebag	Ya
AWS Pemberitahuan Pengguna	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Kontak Pemberitahuan Pengguna	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Langganan Pengguna	Ya	Tida	Tida	Tida	Ya	Tidak
Akses Terverifikasi AWS	Ya	Tida	Tida	Tida	Ya	Tidak
Izin Terverifikasi Amazon	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Virtual Private Cloud (AmazonVPC)	Ya	Sebag (Info)	Sebag (Info)	Ya	Ya	Sebagian (Info)
VPCKisi Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Layanan Amazon VPC Lattice	Ya	Ya	Tida	Tida	Ya	Tidak
AWS WAF	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS WAF Klasik	Ya	Ya	Tida	Ya	Ya	Ya
AWS WAF Regional	Ya	Ya	Tida	Ya	Ya	Ya
AWS Well-Architected Tool	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Wickr	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon WorkDocs	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon WorkMail	Ya	Ya	Tida	Ya	Ya	Ya
Alur WorkMail Pesan Amazon	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon WorkSpaces	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Browser WorkSpaces Aman Amazon	Ya	Ya	Tida	Ya	Ya	Ya
Klien WorkSpaces Tipis Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
AWS X-Ray	Ya	Seba (Info)	Tida	Sebag (Info)	Ya	Tidak

Informasi lain

Amazon CloudFront

CloudFront tidak memiliki peran terkait layanan, tetapi Lambda @Edge memilikinya. Untuk informasi selengkapnya, lihat Peran Tertaut Layanan untuk Lambda @Edge di Panduan Pengembang Amazon. CloudFront

AWS CloudTrail

CloudTrail mendukung kebijakan berbasis sumber daya hanya pada CloudTrail saluran yang digunakan untuk integrasi CloudTrail Lake dengan sumber acara di luar. AWS

Amazon CloudWatch

CloudWatch peran terkait layanan tidak dapat dibuat menggunakan AWS Management Console, dan hanya mendukung fitur Tindakan Alarm.

AWS CodeBuild

CodeBuild mendukung berbagi sumber daya lintas akun menggunakan AWS RAM.

CodeBuild mendukung ABAC tindakan berbasis proyek.

AWS Config

AWS Config mendukung izin tingkat sumber daya untuk agregasi dan Aturan data multi-wilayah multi-akun. AWS Config Untuk daftar sumber daya yang didukung, lihat bagian Agregasi Data Multi-Wilayah Multi-Akun dan bagian AWS Config Aturan pada Panduan.AWS Config API

AWS Database Migration Service

Anda dapat membuat dan mengubah kebijakan yang dilampirkan ke kunci AWS KMS enkripsi yang Anda buat untuk mengenkripsi data yang dimigrasi ke titik akhir target yang didukung. Titik akhir target yang didukung termasuk Amazon Redshift dan Amazon S3. Untuk informasi selengkapnya, lihat Membuat dan Menggunakan AWS KMS Kunci untuk Mengenkripsi Data Target Amazon Redshift dan AWS KMS Membuat Kunci untuk Mengenkripsi Objek Target Amazon S3 di Panduan Pengguna.AWS Database Migration Service

Amazon Elastic Compute Cloud

Peran EC2 terkait layanan Amazon hanya dapat digunakan untuk fitur berikut: Permintaan Instans Spot, Permintaan Armada Spot, Armada Amazon EC2, dan Peluncuran cepat untuk instans Windows.

Amazon Elastic Container Service

Hanya beberapa ECS tindakan Amazon yang mendukung izin tingkat sumber daya.

AWS Elemental MediaPackage

MediaPackage mendukung peran terkait layanan untuk mempublikasikan log akses pelanggan ke CloudWatch tetapi tidak untuk tindakan lainAPI.

AWS Identity and Access Management

IAMHanya mendukung satu jenis kebijakan berbasis sumber daya yang disebut kebijakan kepercayaan peran, yang melekat pada peran. IAM Untuk informasi selengkapnya, lihat Berikan izin pengguna untuk beralih peran.

IAMmendukung kontrol akses berbasis tag untuk sebagian besar IAM sumber daya. Untuk informasi selengkapnya, lihat Tag untuk AWS Identity and Access Management sumber daya.

Hanya beberapa API tindakan untuk yang IAM dapat dipanggil dengan kredensil sementara. Untuk informasi selengkapnya, lihat Membandingkan API opsi Anda.

AWS IoT

Perangkat yang terhubung ke AWS IoT diautentikasi dengan menggunakan sertifikat X.509 atau menggunakan Identitas Amazon Cognito. Anda dapat melampirkan AWS IoT kebijakan ke sertifikat X.509 atau Identitas Amazon Cognito untuk mengontrol apa yang diizinkan oleh perangkat. Untuk informasi selengkapnya, lihat Keamanan dan Identitas untuk AWS IoT dalam Panduan Developer AWS IoT .

AWS Lambda

Lambda mendukung kontrol akses berbasis atribut (ABAC) untuk API tindakan yang menggunakan fungsi Lambda sebagai sumber daya yang diperlukan. Lapisan, pemetaan sumber peristiwa, dan sumber konfigurasi penandatanganan kode tidak didukung.

Lambda tidak memiliki peran terkait layanan, tetapi Lambda @Edge memilikinya. Untuk informasi selengkapnya, lihat Peran Tertaut Layanan untuk Lambda @Edge di Panduan Pengembang Amazon. CloudFront

Amazon Lightsail

Lightsail sebagian mendukung izin tingkat sumber daya dan. ABAC Untuk informasi selengkapnya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon Lightsail.

Amazon Managed Streaming for Apache Kafka () MSK

Anda dapat melampirkan kebijakan klaster ke MSK klaster Amazon yang telah dikonfigurasi untuk VPCmulti-konektivitas.

AWS Network Manager

AWS Cloud WAN juga mendukung peran terkait layanan. Untuk informasi selengkapnya, lihat Peran WAN terkait layanan AWS Cloud di Panduan Amazon VPC AWS Cloud WAN.

Amazon Relational Database Service

Amazon Aurora adalah mesin database relasional yang dikelola sepenuhnya yang kompatibel dengan My SQL dan Postgre. SQL Anda dapat memilih Aurora My atau SQL Aurora Postgre

SQL sebagai opsi mesin DB saat menyiapkan server database baru melalui Amazon. RDS Untuk informasi selengkapnya, lihat Manajemen identitas dan akses untuk Amazon Aurora di Panduan Pengguna Amazon Aurora.

Amazon Rekognition

Kebijakan berbasis sumber daya hanya didukung untuk menyalin model Label Kustom Rekognition Amazon.

AWS Resource Groups

Pengguna dapat mengambil peran dengan kebijakan yang memungkinkan operasi Resource Groups.

Amazon SageMaker

Peran terkait layanan saat ini tersedia untuk pekerjaan SageMaker Studio dan SageMaker pelatihan.

AWS Security Token Service

AWS STS tidak memiliki “sumber daya,” tetapi memungkinkan pembatasan akses dengan cara yang mirip dengan pengguna. Untuk informasi selengkapnya, lihat Menolak akses ke kredensial keamanan sementara menurut nama.

Hanya beberapa API operasi untuk panggilan AWS STS dukungan dengan kredensil sementara. Untuk informasi selengkapnya, lihat Membandingkan API opsi Anda.

Layanan Email Sederhana Amazon

Anda hanya dapat menggunakan izin tingkat sumber daya dalam pernyataan kebijakan yang merujuk pada tindakan yang terkait dengan pengiriman email, seperti atau. `ses:SendEmail` `ses:SendRawEmail` Untuk pernyataan kebijakan yang mengacu pada tindakan lainnya, elemen Sumber daya hanya dapat berisi `*`.

Hanya Amazon yang SES API mendukung kredensil keamanan sementara. SESSMTPAntarmuka Amazon tidak mendukung SMTP kredensil yang berasal dari kredensil keamanan sementara.

Amazon Simple Storage Service

Amazon S3 mendukung otorisasi berbasis tag hanya untuk sumber daya objek.

Amazon S3 mendukung peran terkait layanan untuk Lensa Penyimpanan Amazon S3.

AWS Trusted Advisor

APIAkses ke Trusted Advisor adalah melalui AWS Support API dan dikendalikan oleh AWS Support IAM kebijakan.

Amazon Virtual Private Cloud

Dalam kebijakan IAM pengguna, Anda tidak dapat membatasi izin ke titik akhir Amazon VPC tertentu. Setiap `Action` elemen yang mencakup `ec2:*VpcEndpoint*` atau `ec2:DescribePrefixLists` API tindakan harus menentukan `"Resource": "*"`". Untuk informasi selengkapnya, lihat Identitas dan manajemen akses untuk layanan VPC titik VPC akhir dan titik akhir di Panduan.AWS PrivateLink

Amazon VPC mendukung melampirkan kebijakan sumber daya tunggal ke VPC titik akhir untuk membatasi apa yang dapat diakses melalui titik akhir tersebut. Untuk informasi selengkapnya tentang penggunaan kebijakan berbasis sumber daya untuk mengontrol akses ke sumber daya dari VPC titik akhir Amazon tertentu, lihat Mengontrol akses ke layanan menggunakan kebijakan titik akhir di Panduan.AWS PrivateLink

Amazon VPC tidak memiliki peran terkait layanan, tetapi AWS Transit Gateway melakukannya. Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk gateway transit di Panduan Amazon VPC AWS Transit Gateway .

AWS X-Ray

X-Ray tidak mendukung izin tingkat sumber daya untuk semua tindakan.

X-Ray mendukung kontrol akses berbasis tag untuk grup dan aturan pengambilan sampel.

AWS Signature Version 4 untuk API permintaan

> **Important**
>
> Jika Anda menggunakan alat AWS SDK (lihat Contoh Kode dan Pustaka) atau AWS Command Line Interface (AWS CLI) untuk mengirim API permintaan AWS, Anda dapat melewati proses tanda tangan, karena CLI klien SDK dan mengautentikasi permintaan Anda dengan menggunakan kunci akses yang Anda berikan. Kecuali Anda memiliki alasan yang baik untuk tidak melakukannya, kami sarankan Anda selalu menggunakan SDK atauCLI.

Di Wilayah yang mendukung beberapa versi tanda tangan, menandatangani permintaan secara manual berarti Anda harus menentukan versi tanda tangan yang akan digunakan. Saat Anda menyediakan permintaan ke Titik Akses Multi-Wilayah, SDKs dan CLI secara otomatis beralih menggunakan Signature Version 4A tanpa konfigurasi tambahan.

Informasi otentikasi yang Anda kirim dalam permintaan harus menyertakan tanda tangan. AWS Sigv4 (SigV4) adalah protokol AWS penandatanganan untuk menambahkan informasi otentikasi ke permintaan. AWS API

Anda tidak menggunakan kunci akses rahasia Anda untuk menandatangani API permintaan. Sebagai gantinya, Anda menggunakan proses penandatanganan SiGv4. Permintaan penandatanganan melibatkan:

1. Membuat permintaan kanonik berdasarkan detail permintaan.
2. Menghitung tanda tangan menggunakan AWS kredensi Anda.
3. Menambahkan tanda tangan ini ke permintaan sebagai header Otorisasi.

AWS kemudian mereplikasi proses ini dan memverifikasi tanda tangan, memberikan atau menolak akses yang sesuai.

SigV4 simetris mengharuskan Anda untuk mendapatkan kunci yang dicakup ke satu AWS layanan, di satu AWS wilayah, pada hari tertentu. Ini membuat kunci dan tanda tangan yang dihitung berbeda untuk setiap wilayah, artinya Anda harus mengetahui wilayah yang ditakdirkan untuk tanda tangan tersebut.

Asymmetric Signature Version 4 (Sigv4a) adalah ekstensi yang mendukung penandatanganan dengan algoritma baru, dan menghasilkan tanda tangan individual yang dapat diverifikasi di lebih dari satu wilayah. AWS Dengan Sigv4a, Anda dapat menandatangani permintaan untuk beberapa wilayah, dengan perutean dan failover yang mulus antar wilayah. Saat Anda menggunakan AWS SDK atau AWS CLI untuk memanggil fungsionalitas yang memerlukan penandatanganan multi-wilayah, jenis tanda tangan secara otomatis diubah untuk menggunakan Sigv4a. Untuk detailnya, lihat Cara kerja AWS Sigv4a.

Bagaimana AWS SiGv4 bekerja

Langkah-langkah berikut menjelaskan proses umum komputasi tanda tangan dengan SiGv4:

1. String yang akan ditandatangani tergantung pada jenis permintaan. Misalnya, saat Anda menggunakan header HTTP Otorisasi atau parameter kueri untuk otentikasi, Anda menggunakan kombinasi elemen permintaan untuk membuat string yang akan ditandatangani. Untuk HTTP POST permintaan, `POST` kebijakan dalam permintaan adalah string yang Anda tandatangani.
2. Kunci penandatanganan adalah serangkaian perhitungan, dengan hasil dari setiap langkah dimasukkan ke langkah berikutnya. Langkah terakhir adalah kunci penandatanganan.
3. Ketika sebuah AWS layanan menerima permintaan yang diautentikasi, ia membuat ulang tanda tangan menggunakan informasi otentikasi yang terkandung dalam permintaan. Jika tanda tangan cocok, layanan memproses permintaan. Kalau tidak, itu menolak permintaan.

Untuk informasi selengkapnya, lihat Elemen tanda tangan AWS API permintaan.

Cara kerja AWS Sigv4a

Sigv4a menggunakan tanda tangan asimetris berdasarkan kriptografi kunci publik-pribadi. SIGv4amelalui proses derivasi kredensial cakupan serupa sepertiSIGv4, kecuali Sigv4a menggunakan kunci yang sama untuk menandatangani semua permintaan tanpa perlu mendapatkan kunci penandatanganan yang berbeda berdasarkan tanggal, layanan, dan wilayah. Sebuah Elliptic Curve Digital Signature Algorithm (ECDSA) keypair dapat diturunkan dari kunci akses rahasia yang ada AWS .

Sistem ini menggunakan kriptografi asimetris untuk memverifikasi tanda tangan multi-wilayah, sehingga AWS hanya perlu menyimpan kunci publik Anda. Kunci publik bukan rahasia dan tidak dapat digunakan untuk menandatangani permintaan. Tanda tangan asimetris diperlukan untuk API permintaan multi-wilayah, seperti dengan Titik Akses Multi-Wilayah Amazon S3.

Langkah-langkah berikut menjelaskan proses umum komputasi tanda tangan dengan Sigv4a:

1. String yang akan ditandatangani tergantung pada jenis permintaan. Misalnya, saat Anda menggunakan header HTTP Otorisasi atau parameter kueri untuk otentikasi, Anda menggunakan kombinasi elemen permintaan untuk membuat string yang akan ditandatangani. Untuk HTTP POST permintaan, `POST` kebijakan dalam permintaan adalah string yang Anda tandatangani.
2. Kunci penandatanganan berasal dari kunci akses AWS rahasia melalui serangkaian perhitungan, dengan hasil dari setiap langkah dimasukkan ke langkah berikutnya. Langkah terakhir menghasilkan keypair.
3. Ketika AWS layanan menerima permintaan yang ditandatangani dengan Sigv4a, AWS verifikasi tanda tangan hanya menggunakan separuh publik dari keypair. Jika tanda tangan valid,

permintaan diautentikasi dan layanan memproses permintaan. Permintaan dengan tanda tangan yang tidak valid ditolak.

Untuk informasi selengkapnya tentang Sigv4a untuk API permintaan Multi-wilayah, lihat proyek sigv4 di. a-signing-examples GitHub

Kapan menandatangani permintaan

Saat Anda menulis kode khusus yang mengirimkan API permintaan AWS, Anda harus menyertakan kode yang menandatangani permintaan. Anda dapat menulis kode khusus karena:

- Anda bekerja dengan bahasa pemrograman yang tidak ada AWS SDK.
- Anda memerlukan kontrol penuh atas bagaimana permintaan dikirim AWS.

Sementara API permintaan mengautentikasi akses dengan AWS SigV4, AWS SDKs dan AWS CLI mengautentikasi permintaan Anda dengan menggunakan kunci akses yang Anda berikan. Untuk informasi selengkapnya tentang autentikasi dengan AWS SDKs dan AWS CLI, lihatSumber daya tambahan.

Mengapa permintaan ditandatangani

Proses penandatanganan membantu mengamankan permintaan dengan cara berikut:

- Verifikasi identitas pemohon

  Permintaan yang diautentikasi memerlukan tanda tangan yang Anda buat dengan menggunakan kunci akses Anda (ID kunci akses, kunci akses rahasia). Jika Anda menggunakan kredenal keamanan sementara, perhitungan tanda tangan juga memerlukan token keamanan. Untuk informasi selengkapnya, lihat akses terprogram kredensial AWS keamanan.
- Lindungi data dalam perjalanan

  Untuk mencegah gangguan pada permintaan saat sedang transit, beberapa elemen permintaan digunakan untuk menghitung hash (intisari) permintaan, dan nilai hash yang dihasilkan disertakan sebagai bagian dari permintaan. Ketika Layanan AWS menerima permintaan, ia menggunakan informasi yang sama untuk menghitung hash dan mencocokkannya dengan nilai hash dalam permintaan Anda. Jika nilainya tidak cocok, AWS tolak permintaan tersebut.
- Lindungi dari potensi serangan replay

Dalam kebanyakan kasus, permintaan harus mencapai AWS dalam waktu lima menit dari cap waktu dalam permintaan. Kalau tidak, AWS menolak permintaan itu.

AWS SigV4 dapat diekspresikan dalam header HTTP Otorisasi atau sebagai string kueri di. URL Untuk informasi selengkapnya, lihat Metode otentikasi.

Sumber daya tambahan

- Untuk informasi selengkapnya tentang proses penandatanganan SiGv4 untuk berbagai layanan, lihat. Minta contoh tanda tangan
- Untuk mengonfigurasi kredensional untuk akses terprogram AWS CLI, lihat Otentikasi dan akses kredensional di Panduan Pengguna Antarmuka Baris AWS Perintah.
- AWS SDKsTermasuk kode sumber GitHub untuk menandatangani AWS API permintaan. Untuk contoh kode, lihatContoh proyek dalam AWS repositori sampel.
  - AWS SDK for .NET — AWS4Signer.cs
  - AWS SDK for C++ — AWSAuthV4Signer.cpp
  - AWS SDK for Go — v4.go
  - AWS SDK for Java — BaseAws4Signer.java
  - AWS SDK for JavaScript - v4.js
  - AWS SDK for PHP — SignatureV4.php
  - AWS SDK for Python (Boto) - signers.py
  - AWS SDK for Ruby — penandatangan.rb

Elemen tanda tangan AWS API permintaan

> **Important**
>
> Kecuali Anda menggunakan AWS SDKs atauCLI, Anda harus menulis kode untuk menghitung tanda tangan yang memberikan informasi otentikasi dalam permintaan Anda. Perhitungan AWS tanda tangan di Signature Version 4 dapat menjadi usaha yang rumit, dan kami menyarankan Anda menggunakan AWS SDKs atau CLI bila memungkinkan.

Setiap HTTPS permintaanHTTP/yang menggunakan penandatanganan Signature Version 4 harus berisi elemen-elemen ini.

Elemen

- Spesifikasi titik akhir
- Tindakan
- Parameter tindakan
- Tanggal
- Informasi otentikasi

Spesifikasi titik akhir

Menentukan DNS nama titik akhir yang Anda kirim permintaan. Nama ini biasanya berisi kode layanan dan Wilayah. Misalnya, titik akhir untuk Amazon DynamoDB di Wilayah `us-east-1` adalah. `dynamodb.us-east-1.amazonaws.com`

Untuk permintaan HTTP /1.1, Anda harus menyertakan `Host` header. Untuk permintaan HTTP /2, Anda dapat menyertakan `:authority` header atau `Host` header. Gunakan hanya `:authority` header untuk kepatuhan dengan spesifikasi HTTP /2. Tidak semua layanan mendukung HTTP /2 permintaan.

Untuk titik akhir yang didukung oleh setiap layanan, lihat Titik akhir layanan dan kuota di. Referensi Umum AWS

Tindakan

Menentukan API tindakan untuk layanan. Misalnya, tindakan `CreateTable` DynamoDB atau tindakan Amazon. EC2 `DescribeInstances`

Untuk tindakan yang didukung oleh setiap layanan, lihat Referensi Otorisasi Layanan.

Parameter tindakan

Menentukan parameter untuk tindakan yang ditentukan dalam permintaan. Setiap AWS API tindakan memiliki seperangkat parameter yang diperlukan dan opsional. APIVersi ini biasanya merupakan parameter yang diperlukan.

Untuk parameter yang didukung oleh API tindakan, lihat API Referensi untuk layanan.

Tanggal

Menentukan tanggal dan waktu permintaan. Menyertakan tanggal dan waktu dalam permintaan membantu mencegah pihak ketiga mencegat permintaan Anda dan mengirimkannya kembali nanti. Tanggal yang Anda tentukan dalam lingkup kredensi harus sesuai dengan tanggal permintaan Anda.

Cap waktu harus dalam UTC dan menggunakan format ISO 8601 berikut: T Z. YYYYMMDDHHMMSS Misalnya, `20220830T123600Z`. Jangan sertakan milidetik dalam stempel waktu.

Anda dapat menggunakan `date` atau `x-amz-date` header, atau menyertakan `x-amz-date` sebagai parameter kueri. Jika kita tidak dapat menemukan `x-amz-date` header, maka kita mencari `date` header.

Informasi otentikasi

Setiap permintaan yang Anda kirim harus menyertakan informasi berikut. AWS menggunakan informasi ini untuk memastikan validitas dan keaslian permintaan.

- Algoritma — Gunakan `AWS 4-HMAC-SHA256` untuk menentukan Signature Version 4 dengan algoritma `HMAC-SHA256` hash.
- Credential — String yang terdiri dari ID kunci akses Anda, tanggal dalam YYYYMMDDformat, kode Region, kode layanan, dan string `aws4_request` terminasi, dipisahkan oleh garis miring (/). Kode Region, kode layanan, dan string terminasi harus menggunakan karakter huruf kecil.

  ```
  AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
  ```

- Header yang ditandatangani - HTTP Header untuk disertakan dalam tanda tangan, dipisahkan oleh titik koma (;). Misalnya, `host;x-amz-date`.
- Signature — String yang dikodekan heksadesimal yang mewakili tanda tangan yang dihitung. Anda harus menghitung tanda tangan menggunakan algoritme yang Anda tentukan di parameter `Algorithm`.

Untuk informasi selengkapnya, lihat Metode otentikasi

Metode otentikasi

> **Important**
>
> Kecuali Anda menggunakan AWS SDKs atauCLI, Anda harus menulis kode untuk menghitung tanda tangan yang memberikan informasi otentikasi dalam permintaan Anda. Perhitungan AWS tanda tangan di Signature Version 4 dapat menjadi usaha yang rumit, dan kami menyarankan Anda menggunakan AWS SDKs atau CLI bila memungkinkan.

Anda dapat mengekspresikan informasi otentikasi dengan menggunakan salah satu metode berikut.

HTTPheader otorisasi

HTTP`Authorization`Header adalah metode yang paling umum untuk mengautentikasi permintaan. Semua REST API operasi (kecuali untuk unggahan berbasis browser menggunakan `POST` permintaan) memerlukan header ini. Untuk informasi selengkapnya tentang nilai header otorisasi, dan cara menghitung tanda tangan dan opsi terkait, lihat Mengautentikasi Permintaan: Menggunakan Header Otorisasi (Versi AWS Tanda Tangan 4) di Referensi Amazon S3. API

Berikut ini adalah contoh dari nilai `Authorization` header. Jeda baris ditambahkan ke contoh ini untuk keterbacaan. Dalam kode Anda, header harus berupa string kontinu. Tidak ada koma antara algoritma dan Credential, tetapi elemen lainnya harus dipisahkan dengan koma.

```
Authorization: AWS 4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Tabel berikut menjelaskan berbagai komponen nilai header Otorisasi pada contoh sebelumnya:

Komponen	Deskripsi
Otorisasi	Algoritma yang digunakan untuk menghitung tanda tangan. Anda harus memberikan nilai ini ketika Anda menggunakan AWS Signature Version 4 untuk otentikasi.String menentuka n AWS Signature Version 4 (`AWS 4`) dan algoritma penandatanganan (). `HMAC-SHA256`

Komponen	Deskripsi
Kredensi	ID kunci akses Anda dan informasi ruang lingkup, yang mencakup tanggal, Wilayah, dan layanan yang digunakan untuk menghitung tanda tangan. String ini memiliki bentuk sebagai berikut: `<your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request` Dimana: `<date>` nilai ditentukan menggunak an YYYYMMDD format. `<aws-serv ice>` nilainya adalah s3 saat mengirim permintaan ke Amazon S3.
SignedHeaders	Daftar header permintaan yang dipisahkan titik koma yang Anda gunakan untuk menghitung. `Signature` Daftar ini hanya mencakup nama header, dan nama header harus dalam huruf kecil. Misalnya: host; jangkauan; x-amz-date
Tanda tangan	Tanda tangan 256-bit dinyatakan sebagai 64 karakter heksadesimal huruf kecil. Sebagai contoh: `fe5f80f77d5fa3beca038a248ff 027d0445342fe2855ddc9631766 30326f1024` Perhatikan bahwa perhitungan tanda tangan bervariasi tergantung pada opsi yang Anda pilih untuk mentransfer muatan.

Parameter string kueri

Anda dapat menggunakan string kueri untuk mengekspresikan permintaan sepenuhnya dalam fileURL. Dalam hal ini, Anda menggunakan parameter kueri untuk memberikan informasi permintaan,

termasuk informasi otentikasi. Karena tanda tangan permintaan adalah bagian dariURL, jenis URL ini sering disebut sebagai presignedURL. Anda dapat menggunakan prasesigned URLs to embed link yang dapat diklikHTML, yang dapat berlaku hingga tujuh hari. Untuk informasi selengkapnya, lihat Mengautentikasi Permintaan: Menggunakan Parameter Kueri (Versi AWS Tanda Tangan 4) di Referensi Amazon API S3.

Berikut ini adalah contoh presignedURL. Jeda baris ditambahkan ke contoh ini untuk keterbacaan:

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/us-east-1/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

> **Note**
>
> `X-Amz-Credential`Nilai dalam URL menunjukkan karakter "/" hanya untuk keterbacaan. Dalam prakteknya, itu harus dikodekan sebagai% 2F. Sebagai contoh: `&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request`

Tabel berikut menjelaskan parameter query dalam URL yang menyediakan informasi otentikasi.

Nama parameter string kueri	Deskripsi
X-Amz-Algorithm	Mengidentifikasi versi AWS Signature dan algoritma yang Anda gunakan untuk menghitung tanda tangan.Untuk AWS Signature Version 4, Anda menetapkan nilai parameter ini ke. `AWS4-HMAC-SHA256` String ini mengidentifikasi AWS Signature Version 4 (AWS 4) dan HMAC-SHA256 algorithm (`HMAC-SHA256`).
X-Amz-Credential	Selain ID kunci akses Anda, parameter ini juga menyediakan ruang lingkup (AWS Wilayah dan layanan) yang tanda tangannya valid. Nilai

Nama parameter string kueri	Deskripsi
	ini harus sesuai dengan cakupan yang Anda gunakan dalam perhitungan tanda tangan, yang dibahas di bagian berikut. Bentuk umum untuk nilai parameter ini adalah sebagai berikut: `<your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request` Misalnya: `AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request` Untuk daftar string AWS regional, lihat Titik Akhir Regional di Referensi AWS Umum.
X-Amz-Date	Format tanggal dan waktu harus mengikuti standar ISO 8601, dan harus diformat dengan format. `yyyyMMddTHHmmssZ` Misalnya jika tanggal dan waktu adalah “08/01/2016 15:32:41.982-700" maka pertama-tama harus dikonversi ke UTC (Coordinated Universal Time) dan kemudian diserahkan sebagai “20160801T223241Z”.

Nama parameter string kueri	Deskripsi
X-Amz-Kedaluwarsa	Menyediakan periode waktu, dalam hitungan detik, di mana presigned yang URL dihasilka n valid. Misalnya, 86400 (24 jam). Nilai ini adalah bilangan bulat. Nilai minimum yang dapat Anda tetapkan adalah 1, dan maksimum adalah 604800 (tujuh hari) .Presigned URL dapat berlaku selama maksimal tujuh hari karena kunci penandatanganan yang Anda gunakan dalam perhitungan tanda tangan berlaku hingga tujuh hari.
X-Amz- SignedHeaders	Daftar header yang Anda gunakan untuk menghitung tanda tangan. Header berikut diperlukan dalam perhitungan tanda tangan: • Header HTTP host. • Setiap header x-amz-* yang Anda rencanaka n untuk ditambahkan ke permintaan. Untuk keamanan tambahan, Anda harus menandatangani semua header permintaan yang Anda rencanakan untuk disertakan dalam permintaan Anda.
X-Amz-Signature	Memberikan tanda tangan untuk mengauten tikasi permintaan Anda. Tanda tangan ini harus sesuai dengan tanda tangan yang dihitung oleh layanan; jika tidak, layanan menolak permintaa n tersebut. Sebagai contoh, `733255ef0 22bec3f2a8701cd61d4b371f3f2 8c9f193a1f02279211d48d5193d7` . Perhitungan tanda tangan dijelaskan di bagian berikut.

Nama parameter string kueri	Deskripsi
X-Amz-Security-Token	Parameter kredensi opsional jika menggunakan kredensil yang bersumber dari layanan. STS

Buat AWS API permintaan yang ditandatangani

> **Important**
>
> Jika Anda menggunakan alat AWS SDK (lihat Contoh Kode dan Pustaka) atau AWS Command Line Interface (AWS CLI) untuk mengirim API permintaan AWS, Anda dapat melewati bagian ini karena SDK dan CLI klien mengautentikasi permintaan Anda dengan menggunakan kunci akses yang Anda berikan. Kecuali Anda memiliki alasan yang baik untuk tidak melakukannya, kami sarankan Anda selalu menggunakan SDK atauCLI.
> Di Wilayah yang mendukung beberapa versi tanda tangan, menandatangani permintaan secara manual berarti Anda harus menentukan versi tanda tangan yang digunakan. Saat Anda menyediakan permintaan ke Titik Akses Multi-Wilayah, SDKs dan CLI secara otomatis beralih menggunakan Signature Version 4A tanpa konfigurasi tambahan.

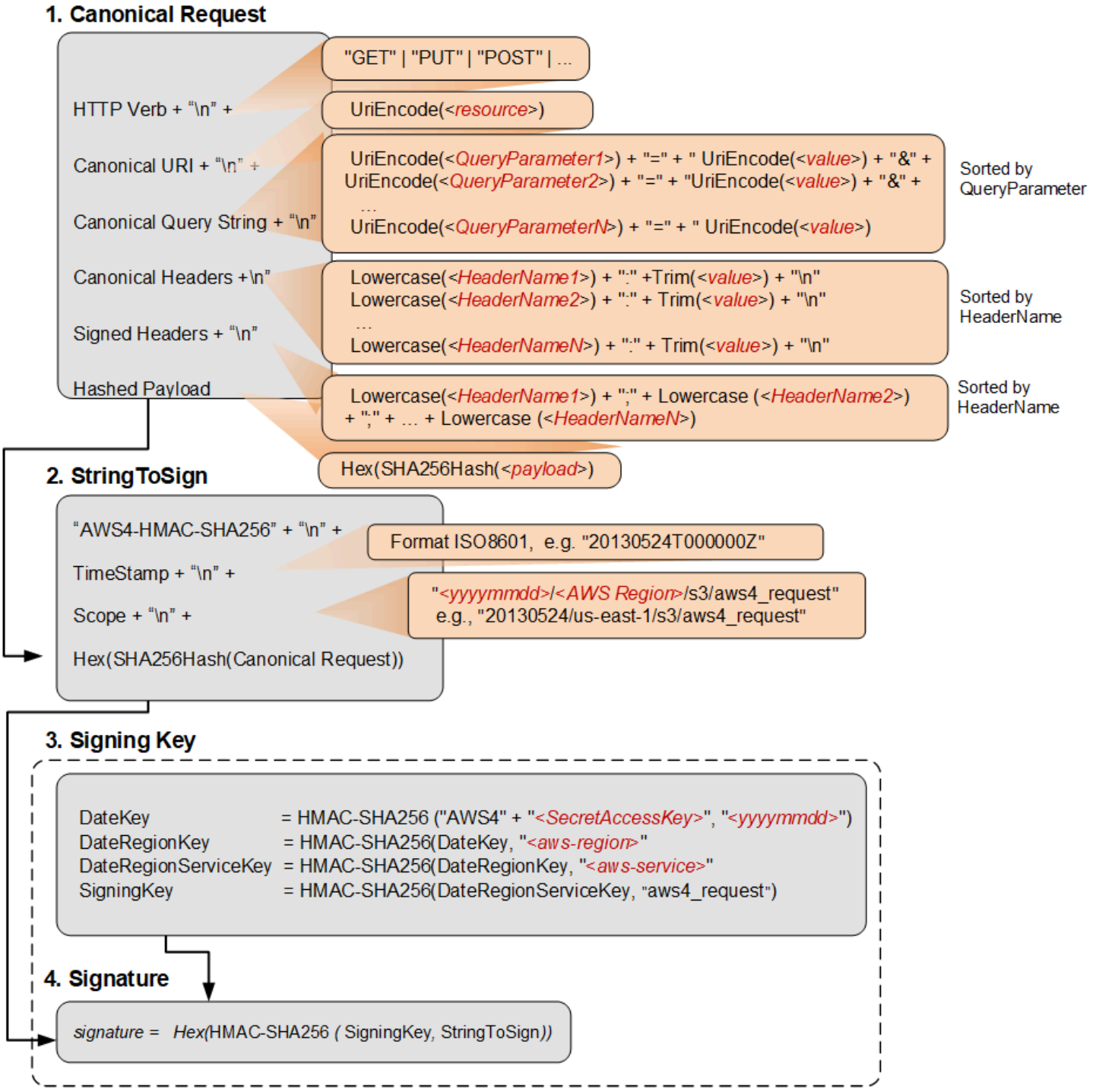
Anda dapat menggunakan protokol penandatanganan AWS SigV4 untuk membuat permintaan permintaan yang ditandatangani. AWS API

1. Membuat permintaan kanonik berdasarkan detail permintaan.
2. Menghitung tanda tangan menggunakan AWS kredensional Anda.
3. Menambahkan tanda tangan ini ke permintaan sebagai header Otorisasi.

AWS kemudian mereplikasi proses ini dan memverifikasi tanda tangan, memberikan atau menolak akses yang sesuai.

Untuk melihat bagaimana Anda dapat menggunakan AWS SiGv4 untuk menandatangani API permintaan, lihat Minta contoh tanda tangan

Diagram berikut menggambarkan proses penandatanganan SigV4, termasuk berbagai komponen string yang Anda buat untuk ditandatangani.



Tabel berikut menjelaskan fungsi yang ditunjukkan dalam diagram. Anda perlu menerapkan kode untuk fungsi-fungsi ini. Untuk informasi selengkapnya, lihat contoh kode di AWS SDKs.

Fungsi	Deskripsi
`Lowercase()`	Ubah string menjadi huruf kecil.
`Hex()`	Pengkodean basis 16 huruf kecil.
`SHA256Hash()`	Secure Hash Algorithm (SHA) fungsi hash kriptografi.
`HMAC-SHA256()`	Menghitung HMAC dengan menggunakan SHA256 algoritma dengan kunci penandata nganan yang disediakan. Ini adalah tanda tangan terakhir.
`Trim()`	Hapus spasi putih di depan atau belakang.
`UriEncode()`	URImengkodekan setiap byte. UriEncode() harus menegakkan aturan berikut: • URImenyandikan setiap byte kecuali karakter tanpa syarat: 'A'-'Z', 'a'-'z', '0'-'9', '-', ' . ',' _ ', dan '~'. • Karakter spasi adalah karakter cadangan dan harus dikodekan sebagai “% 20" (dan bukan sebagai “+”). • Setiap byte URI yang dikodekan dibentuk oleh '%' dan nilai heksadesimal dua digit dari byte. • Huruf dalam nilai heksadesimal harus huruf besar, misalnya “% 1A”. • Encode karakter garis miring maju, '/', di mana-mana kecuali dalam nama kunci objek. Misalnya, jika nama kunci objek adalah`photos/Jan/sample.jpg` , garis miring ke depan dalam nama kunci tidak dikodekan.

Fungsi	Deskripsi
	Important UriEncode Fungsi standar yang disediakan oleh platform pengemban gan Anda mungkin tidak berfungsi karena perbedaan dalam implement asi dan ambiguitas terkait di dasarnyaR FCs. Kami menyarankan Anda menulis UriEncode fungsi kustom Anda sendiri untuk memastikan bahwa pengkodean Anda akan berfungsi. Untuk melihat contoh UriEncode fungsi di Java, lihat Java Utilities di GitHub situs web.

Note

Saat menandatangani permintaan, Anda dapat menggunakan AWS Signature Version 4 atau AWS Signature Version 4A. Perbedaan utama antara keduanya ditentukan oleh bagaimana tanda tangan dihitung. Dengan AWS Signature Version 4A, tanda tangan tidak menyertakan informasi khusus Wilayah dan dihitung menggunakan algoritme. `AWS 4-ECDSA-P256-SHA256`

Menandatangani permintaan dengan kredensyal keamanan sementara

Alih-alih menggunakan kredensyal jangka panjang untuk menandatangani permintaan, Anda dapat menggunakan kredensyal keamanan sementara yang disediakan oleh (). AWS Security Token Service AWS STS

Saat Anda menggunakan kredensyal keamanan sementara, Anda harus menambahkan `X-Amz-Security-Token` ke header Otorisasi atau memasukkannya ke dalam string kueri untuk menahan token sesi. Beberapa layanan mengharuskan Anda `X-Amz-Security-Token` menambahkan permintaan kanonik. Layanan lain hanya mengharuskan Anda menambahkan `X-Amz-Security-`

`Token` di akhir, setelah Anda menghitung tanda tangan. Periksa dokumentasi Layanan AWS untuk masing-masing persyaratan tertentu.

Ringkasan langkah-langkah penandatanganan

Buat permintaan kanonik:

Atur konten permintaan Anda (host, action, header, dll.) Ke dalam format kanonik standar. Permintaan kanonik adalah salah satu input yang digunakan untuk membuat string untuk ditandatangani. Untuk detail tentang membuat permintaan kanonik, lihat. Elemen tanda tangan AWS API permintaan

Buat hash dari permintaan kanonik

Hash permintaan kanonik menggunakan algoritma yang sama yang Anda gunakan untuk membuat hash dari payload. Hash dari permintaan kanonik adalah string karakter heksadesimal huruf kecil.

Buat String untuk Ditandatangani

Buat string untuk ditandatangani dengan permintaan kanonik dan informasi tambahan seperti algoritma, tanggal permintaan, cakupan kredensi, dan hash dari permintaan kanonik.

Turunkan kunci penandatanganan

Lakukan serangkaian operasi hash kunci (HMAC) pada tanggal permintaan, Wilayah, dan layanan, dengan kunci akses AWS rahasia Anda sebagai kunci untuk operasi hashing awal.

Hitung tanda tangan

Lakukan operasi hash berkunci (HMAC) pada string untuk menandatangani menggunakan kunci penandatanganan turunan sebagai kunci hash.

Tambahkan tanda tangan ke permintaan

Tambahkan tanda tangan yang dihitung ke HTTP header atau ke string kueri permintaan.

Buat permintaan kanonik

Untuk membuat permintaan kanonik, gabungkan string berikut, dipisahkan oleh karakter baris baru. Ini membantu memastikan bahwa tanda tangan yang Anda hitung dapat cocok dengan tanda tangan yang AWS menghitung.

```
<HTTPMethod>\n
```

```
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- *HTTPMethod* — HTTP Metode, sepertiGET,, PUTHEAD, danDELETE.
- *CanonicalUri* — Versi URI -encoded dari komponen jalur absolutURI, dimulai dengan / yang mengikuti nama domain dan hingga akhir string atau ke karakter tanda tanya (?) jika Anda memiliki parameter string kueri. Jika jalur absolut kosong, gunakan karakter garis miring maju (/). URIDalam contoh berikut,/amzn-s3-demo-bucket/myphoto.jpg, adalah jalur absolut dan Anda tidak menyandikan / di jalur absolut:

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket/myphoto.jpg
  ```

- *CanonicalQueryString* — Parameter string URI kueri yang dikodekan. Anda URI -encode setiap nama dan nilai satu per satu. Anda juga harus mengurutkan parameter dalam string kueri kanonik menurut abjad dengan nama kunci. Penyortiran terjadi setelah pengkodean. String query dalam URI contoh berikut adalah:

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket?prefix=somePrefix&marker=someMarker&max-keys=2
  ```

  String kueri kanonik adalah sebagai berikut (jeda baris ditambahkan ke contoh ini untuk keterbacaan):

  ```
  UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
  UriEncode("max-keys")+"="+UriEncode("20") + "&" +
  UriEncode("prefix")+"="+UriEncode("somePrefix")
  ```

  Ketika permintaan menargetkan subresource, nilai parameter query yang sesuai akan menjadi string kosong (""). Misalnya, berikut ini URI mengidentifikasi ACL subresource pada bucket: amzn-s3-demo-bucket

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket?acl
  ```

  Dalam hal ini, CanonicalQueryString akan menjadi:

```
UriEncode("acl") + "=" + ""
```

Jika URI tidak menyertakan a?, tidak ada string kueri dalam permintaan, dan Anda mengatur string kueri kanonik ke string kosong ()"". Anda masih perlu menyertakan karakter baris baru ("\n").

- *CanonicalHeaders* — Daftar header permintaan dengan nilainya. Nama header individu dan pasangan nilai dipisahkan oleh karakter baris baru ("\n"). Berikut ini adalah contoh dari CanonicalHeader:

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

CanonicalHeaders daftar harus mencakup yang berikut:

- HTTPhostsundulan.
- Jika `Content-Type` header ada dalam permintaan, Anda harus menambahkannya ke *CanonicalHeaders* daftar.
- Setiap `x-amz-*` header yang Anda rencanakan untuk disertakan dalam permintaan Anda juga harus ditambahkan. Misalnya, jika Anda menggunakan kredensyal keamanan sementara, Anda harus memasukkan `x-amz-security-token` dalam permintaan Anda. Anda harus menambahkan header ini dalam daftar *CanonicalHeaders*.

> **Note**
>
> `x-amz-content-sha256`Header diperlukan untuk permintaan Amazon S3 AWS . Ini menyediakan hash dari payload permintaan. Jika tidak ada payload, Anda harus memberikan hash dari string kosong.

Setiap nama header harus:

- gunakan karakter huruf kecil.
- muncul dalam urutan abjad.
- diikuti oleh titik dua (`:`).

Untuk nilai, Anda harus:

- potong ruang depan atau belakang.
- mengubah ruang berurutan menjadi satu ruang.
- pisahkan nilai untuk header multi-nilai menggunakan koma.
- Anda harus menyertakan header host (HTTP/1.1) atau:authority header (HTTP/2), dan `x-amz-*` header apa pun dalam tanda tangan. Anda dapat secara opsional menyertakan header standar lainnya dalam tanda tangan, seperti tipe konten.

`Trim()`Fungsi `Lowercase()` dan yang digunakan dalam contoh ini dijelaskan di bagian sebelumnya.

Berikut ini adalah contoh `CanonicalHeaders` string. Nama header dalam huruf kecil dan diurutkan.

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> Untuk tujuan menghitung tanda tangan otorisasi, hanya host dan `x-amz-*` header apa pun yang diperlukan; Namun, untuk mencegah gangguan data, Anda harus mempertimbangkan untuk memasukkan semua header dalam perhitungan tanda tangan.

- *`SignedHeaders`* — Daftar nama header permintaan huruf kecil yang diurutkan berdasarkan abjad dan dipisahkan titik koma. Header permintaan dalam daftar adalah header yang sama yang Anda sertakan dalam string. `CanonicalHeaders` Untuk contoh sebelumnya, nilai *`SignedHeaders`* akan menjadi sebagai berikut:

```
host;x-amz-content-sha256;x-amz-date
```

- *`HashedPayload`* — String yang dibuat menggunakan payload di badan HTTP permintaan sebagai input ke fungsi hash. String ini menggunakan karakter heksadesimal huruf kecil.

```
Hex(SHA256Hash(<payload>))
```

Jika tidak ada payload dalam permintaan, Anda menghitung hash dari string kosong, seperti ketika Anda mengambil objek dengan menggunakan GET permintaan, tidak ada apa-apa dalam payload.

```
Hex(SHA256Hash(""))
```

> **Note**
>
> Untuk Amazon S3, sertakan string literal UNSIGNED-PAYLOAD saat membuat permintaan kanonik, dan tetapkan nilai yang sama dengan nilai x-amz-content-sha256 header saat mengirim permintaan.
> Hex(SHA256Hash("UNSIGNED-PAYLOAD"))

Buat hash dari permintaan kanonik

Buat hash (digest) dari permintaan kanonik menggunakan algoritma yang sama yang Anda gunakan untuk membuat hash dari payload. Hash dari permintaan kanonik adalah string karakter heksadesimal huruf kecil.

Buat string untuk ditandatangani

Untuk membuat string untuk ditandatangani, gabungkan string berikut, dipisahkan oleh karakter baris baru. Jangan akhiri string ini dengan karakter baris baru.

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *Algorithm* — Algoritma yang digunakan untuk membuat hash dari permintaan kanonik. Untuk SHA -256, algoritmanya adalahAWS4-HMAC-SHA256.
- *RequestDateTime* — Tanggal dan waktu yang digunakan dalam lingkup kredensi. Nilai ini adalah UTC waktu saat ini dalam format ISO 8601 (misalnya,20130524T000000Z).
- *CredentialScope* — Cakupan kredensi, yang membatasi tanda tangan yang dihasilkan ke Wilayah dan layanan yang ditentukan. String memiliki format berikut: *YYYYMMDD*/*region*/*service*/aws4_permintaan.

- *HashedCanonicalRequest* — Hash dari permintaan kanonik, dihitung pada langkah sebelumnya.

Berikut ini adalah contoh string untuk menandatangani.

```
"AWS4-HMAC-SHA256" + "\n" +
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(SHA256Hash(<CanonicalRequest>))
```

Turunkan kunci penandatanganan

Untuk mendapatkan kunci penandatanganan, lakukan suksesi operasi hash berkunci (HMAC) pada tanggal permintaan, Wilayah, dan layanan, dengan kunci akses AWS rahasia Anda sebagai kunci untuk operasi hashing awal.

Untuk setiap langkah, panggil fungsi hash dengan kunci dan data yang diperlukan. Hasil dari setiap panggilan ke fungsi hash menjadi input untuk panggilan berikutnya ke fungsi hash.

Contoh berikut menunjukkan bagaimana Anda mendapatkan yang `SigningKey` digunakan di bagian selanjutnya dari prosedur ini, menunjukkan urutan input Anda digabungkan dan di-hash. `HMAC-SHA256`adalah fungsi hash yang digunakan untuk hash data seperti yang ditunjukkan.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Input yang dibutuhkan

- `Key`, string yang berisi kunci akses rahasia Anda.
- `Date`, string yang berisi tanggal yang digunakan dalam lingkup kredensi, dalam format YYYYMMDD.
- `Region`, string yang berisi kode Wilayah (misalnya,`us-east-1`).

  Untuk daftar string Region, lihat Titik Akhir Regional di. Referensi Umum AWS
- `Service`, string yang berisi kode layanan (misalnya,`ec2`).
- String untuk menandatangani bahwa Anda membuat pada langkah sebelumnya.

Untuk mendapatkan kunci penandatanganan

1. Concatenate "AWS4" dan kunci akses rahasia. Panggil fungsi hash dengan string gabungan sebagai kunci dan string tanggal sebagai data.

   ```
   DateKey = hash("AWS4" + Key, Date)
   ```

2. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string Wilayah sebagai data.

   ```
   DateRegionKey = hash(kDate, Region)
   ```

3. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string layanan sebagai data.

   Kode layanan ditentukan oleh layanan. Anda dapat menggunakan produk dalam AWS Harga CLI untuk mengembalikan kode layanan untuk suatu layanan.

   ```
   DateRegionServiceKey = hash(kRegion, Service)
   ```

4. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan “aws4_request” sebagai data.

   ```
   SigningKey = hash(kService, "aws4_request")
   ```

Hitung tanda tangan

Setelah Anda mendapatkan kunci penandatanganan, hitung tanda tangan dengan melakukan operasi hash kunci pada string yang akan ditandatangani. Gunakan kunci penandatanganan turunan sebagai kunci hash untuk operasi ini.

Untuk menghitung tanda tangan

1. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string untuk ditandatangani sebagai data. Hasilnya adalah tanda tangan sebagai nilai biner.

   ```
   signature = hash(SigningKey, string-to-sign)
   ```

2. Mengkonversi tanda tangan dari biner ke representasi heksadesimal, dalam karakter huruf kecil.

Tambahkan tanda tangan ke permintaan

Tambahkan tanda tangan yang dihitung ke permintaan Anda.

Example Contoh: Header otorisasi

Contoh berikut menunjukkan `Authorization` header untuk `DescribeInstances` tindakan. Untuk keterbacaan, contoh ini diformat dengan jeda baris. Dalam kode Anda, ini harus berupa string kontinu. Tidak ada koma antara algoritme dan `Credential`. Namun, elemen lainnya harus dipisahkan dengan koma.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

Example Contoh: Permintaan dengan parameter otentikasi dalam string kueri

Contoh berikut menunjukkan kueri untuk `DescribeInstances` tindakan yang menyertakan informasi otentikasi. Untuk keterbacaan, contoh ini diformat dengan jeda baris dan tidak URL dikodekan. Dalam kode Anda, string kueri harus berupa string kontinu yang URL dikodekan.

```
https://ec2.amazonaws.com/?
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

Minta contoh tanda tangan

Contoh permintaan AWS penandatanganan berikut menunjukkan kepada Anda bagaimana Anda dapat menggunakan SigV4 untuk menandatangani permintaan yang dikirim tanpa alat AWS SDK atau baris AWS perintah.

Unggahan Amazon S3 berbasis browser menggunakan HTTP POST

Permintaan Autentikasi: Unggahan Berbasis Browser menjelaskan tanda tangan dan informasi relevan yang digunakan Amazon S3 untuk menghitung tanda tangan setelah menerima permintaan.

Contoh: Unggahan Berbasis Browser menggunakan HTTP POST (Menggunakan AWS Tanda Tangan Versi 4) memberikan informasi lebih lanjut dengan POST kebijakan sampel dan formulir yang dapat Anda gunakan untuk mengunggah file. Contoh kebijakan dan kredenal fiktif menunjukkan alur kerja dan tanda tangan yang dihasilkan serta hash kebijakan.

VPCPermintaan otentikasi kisi

Contoh untuk permintaan terautentikasi Signature Version 4 (SigV4) memberikan contoh Python dan Java yang menunjukkan bagaimana Anda dapat melakukan penandatanganan permintaan dengan dan tanpa pencegat khusus.

Menggunakan Signature Versi 4 dengan Amazon Translate

Terjemahan Langsung di Metaverse menunjukkan cara membuat aplikasi yang menghasilkan solusi terjemahan mendekati waktu nyata. Solusi speech-to-speech penerjemah ini menggunakan AWS SiGv4 dalam pengkodean aliran peristiwa untuk menghasilkan transkripsi waktu nyata.

Menggunakan Signature Versi 4 dengan Neptunus

Contoh: Menghubungkan ke Neptunus Menggunakan Python dengan Signature Version 4 Signature Menandatangani menunjukkan cara membuat permintaan yang ditandatangani ke Neptunus menggunakan Python. Contoh ini mencakup variasi untuk menggunakan kunci akses atau kredensi sementara.

Menandatangani HTTP permintaan ke S3 Glacier

Contoh Perhitungan Tanda Tangan untuk Streaming API memandu Anda melalui detail pembuatan tanda tangan untuk Unggah Arsip (POSTarsip), salah satu dari dua streaming APIs di S3 Glacier.

Membuat HTTP Permintaan ke Amazon SWF

Membuat HTTP Permintaan ke Amazon SWF menunjukkan konten header untuk JSON permintaan ke AmazonSWF.

Perhitungan tanda tangan untuk streaming APIs di Amazon OpenSearch Service

Menandatangani permintaan pencarian OpenSearch Layanan Amazon dengan AWS SDK untuk PHP Versi 3 mencakup contoh cara mengirim HTTP permintaan yang ditandatangani ke OpenSearch Layanan Amazon.

Contoh proyek dalam AWS repositori sampel

Contoh proyek berikut menunjukkan cara menandatangani permintaan untuk membuat API permintaan Istirahat ke AWS layanan dengan bahasa umum seperti Python, Node.js, Java, C #, Go dan Rust.

Proyek Signature Version 4a

Proyek sigv4-signing-examples memberikan contoh cara menandatangani permintaan dengan SiGv4 untuk membuat API permintaan Istirahat dengan bahasa Layanan AWS umum seperti Python, Node.js, Java, C #, Go dan Rust.

a-signing-examplesProyek sigv4 memberikan contoh untuk menandatangani API permintaan Multi-wilayah, misalnya Titik Akses Multi-Wilayah di Amazon S3.

Publikasikan ke AWS IoT Core

Kode Python untuk dipublikasikan AWS IoT Core menggunakan HTTPs protokol memberikan panduan tentang cara mempublikasikan pesan untuk AWS IoT Core menggunakan HTTPS protokol dan otentikasi AWS SigV4. Ini memiliki dua implementasi referensi - satu di Python dan lainnya di. NodeJs

Aplikasi .Net Framework untuk mempublikasikan AWS IoT Core menggunakan HTTPs protokol memberikan panduan tentang cara mempublikasikan pesan untuk AWS IoT Core menggunakan HTTPS protokol dan otentikasi AWS SigV4. Proyek ini juga mencakup a. NETimplementasi setara inti.

Memecahkan masalah Tanda Tangan Versi 4 penandatanganan untuk permintaan AWS API

> **Important**
>
> Kecuali Anda menggunakan AWS SDKs atauCLI, Anda harus menulis kode untuk menghitung tanda tangan yang memberikan informasi otentikasi dalam permintaan Anda. Perhitungan tanda tangan SiGv4 dapat menjadi usaha yang rumit, dan kami menyarankan Anda menggunakan AWS SDKs atau CLI bila memungkinkan.

Saat Anda mengembangkan kode yang membuat permintaan yang ditandatangani, Anda mungkin menerima HTTP 403 `SignatureDoesNotMatch` dari Layanan AWS. Kesalahan ini berarti bahwa

nilai tanda tangan dalam HTTP permintaan Anda AWS tidak cocok dengan tanda tangan yang Layanan AWS dihitung. HTTP`Unauthorized`Kesalahan 401 kembali ketika izin tidak mengizinkan pemanggil untuk membuat permintaan.

APIpermintaan mungkin mengembalikan kesalahan jika:

- APIPermintaan tidak ditandatangani dan API permintaan menggunakan IAM otentikasi.
- IAMKredensi yang digunakan untuk menandatangani permintaan tidak benar atau tidak memiliki izin untuk memanggil. API
- Tanda tangan API permintaan yang ditandatangani tidak cocok dengan tanda tangan yang dihitung oleh AWS layanan.
- Header API permintaan tidak benar.

> **Note**
>
> Perbarui protokol penandatanganan Anda dari AWS Signature versi 2 (SigV2) ke AWS Signature versi 4 (SigV4) sebelum menjelajahi solusi kesalahan lainnya. Layanan, seperti Amazon S3, dan Wilayah tidak lagi mendukung penandatanganan SigV2.

Kemungkinan penyebab

- Kesalahan kredensi
- Permintaan kanonik dan kesalahan string penandatanganan
- Kesalahan cakupan kredensi
- Kesalahan penandatanganan kunci

Kesalahan kredensi

Pastikan bahwa API permintaan ditandatangani dengan SiGv4. Jika API permintaan tidak ditandatangani, maka Anda mungkin menerima kesalahan:`Missing Authentication Token`. Tambahkan tanda tangan yang hilang dan kirim ulang permintaan.

Verifikasi bahwa kredensi otentikasi untuk kunci akses dan kunci rahasia sudah benar. Jika kunci akses salah, maka Anda mungkin menerima kesalahan:`Unauthorized`. Pastikan entitas yang digunakan untuk menandatangani permintaan berwenang untuk membuat permintaan. Untuk detailnya, lihat Memecahkan masalah akses ditolak pesan kesalahan.

Permintaan kanonik dan kesalahan string penandatanganan

Jika Anda salah menghitung permintaan kanonik di Buat hash dari permintaan kanonik atauBuat string untuk ditandatangani, langkah verifikasi tanda tangan yang dilakukan oleh layanan gagal dengan pesan kesalahan:

```
The request signature we calculated does not match the signature you provided
```

Ketika AWS layanan menerima permintaan yang ditandatangani, itu menghitung ulang tanda tangan. Jika ada perbedaan nilai, maka tanda tangan tidak cocok. Bandingkan permintaan dan string kanonik dengan permintaan yang ditandatangani dengan nilai dalam pesan kesalahan. Ubah proses penandatanganan jika ada perbedaan.

> **Note**
>
> Anda juga dapat memverifikasi bahwa Anda tidak mengirim permintaan melalui proxy yang mengubah header atau permintaan.

Example Contoh permintaan kanonik

```
GET                                                    -------- HTTP method
/                                                      -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
                                                       -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                          -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com    -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                           -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501       -------- Hash
 of the payload
```

Untuk memverifikasi bahwa kunci rahasia cocok dengan ID kunci akses, Anda dapat mengujinya dengan implementasi kerja yang diketahui. Misalnya, gunakan AWS SDK atau AWS CLI untuk membuat permintaan AWS.

APIpermintaan header

Ketika header otorisasi kosong, kunci kredensi atau tanda tangan hilang atau salah, header tidak dimulai dengan nama algoritme, atau pasangan nilai kunci tidak menyertakan tanda sama dengan, Anda menerima salah satu kesalahan berikut:

- Header otorisasi tidak boleh kosong.
- Header otorisasi membutuhkan parameter 'Kredensial'.
- Header otorisasi membutuhkan parameter 'Tanda tangan'.
- Tanda tangan berisi pasangan key=value yang tidak valid (tidak ada tanda sama) di header Otorisasi.

Pastikan bahwa header otorisasi SigV4 yang Anda tambahkan Hitung tanda tangan menyertakan kunci kredensi yang benar, dan juga menyertakan tanggal permintaan menggunakan HTTP Tanggal atau header. `x-amz-date`

Jika Anda menerima IncompleteSignatureException kesalahan dan konstruksi tanda tangan sudah benar, Anda dapat memverifikasi bahwa header otorisasi tidak dimodifikasi dalam perjalanan ke Layanan AWS dengan menghitung SHA -256 hash dan pengkodean B64 dari header otorisasi dalam permintaan sisi klien Anda.

1. Dapatkan header otorisasi yang Anda kirim dalam permintaan. Header otorisasi Anda tampak mirip dengan contoh berikut:

   ```
   Authorization: AWS4-HMAC-SHA256
   Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
   SignedHeaders=host;range;x-amz-date,
   Signature=example-generated-signature
   ```

2. Hitung hash SHA -256 dari header otorisasi.

   ```
   hashSHA256(rawAuthorizationHeader) = hashedAuthorizationHeader
   ```

3. Encode header otorisasi hash ke format Base64.

   ```
   base64(hashedAuthorizationHeader) = encodedHashedAuthorizationHeader
   ```

4. Bandingkan string yang di-hash dan dikodekan yang baru saja Anda hitung terhadap string yang Anda terima dalam pesan kesalahan Anda. Pesan kesalahan Anda harus mirip dengan contoh berikut:

```
com.amazon.coral.service#IncompleteSignatureException:
The signature contains an in-valid key=value pair (missing equal-sign)
in Authorization header (hashed with SHA-256 and encoded with Base64):
'9c574f83b4b950926da4a99c2b43418b3db8d97d571b5e18dd0e4f3c3ed1ed2c'.
```

- Jika kedua hash berbeda, maka beberapa bagian dari header otorisasi berubah saat transit. Perubahan ini bisa disebabkan oleh penangan jaringan atau klien Anda yang melampirkan header yang ditandatangani atau mengubah header otorisasi dengan cara tertentu.
- Jika kedua hash cocok, header otorisasi yang Anda kirim dalam permintaan cocok dengan apa yang AWS diterima. Tinjau pesan galat yang Anda terima untuk menentukan apakah masalahnya adalah hasil dari kredensi atau tanda tangan yang salah. Kesalahan ini tercakup di bagian lain di halaman ini.

Kesalahan cakupan kredensi

Cakupan kredensi yang Anda buat Buat string untuk ditandatangani membatasi tanda tangan ke tanggal, Wilayah, dan layanan tertentu. String ini memiliki format berikut:

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> Jika Anda menggunakan Sigv4a, Wilayah tidak termasuk dalam cakupan kredensi.

Tanggal

Jika cakupan kredensi tidak menentukan tanggal yang sama dengan x-amz-date header, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

Jika permintaan menentukan waktu di masa mendatang, langkah verifikasi tanda tangan gagal dengan pesan galat berikut:

```
Signature not yet current: date is still later than date
```

Jika permintaan telah kedaluwarsa, langkah verifikasi tanda tangan gagal dengan pesan galat berikut:

```
Signature expired: date is now earlier than date
```

Wilayah

Jika cakupan kredensi tidak menentukan Wilayah yang sama dengan permintaan, langkah verifikasi tanda tangan gagal dengan pesan galat berikut:

```
Credential should be scoped to a valid Region, not region-code
```

Layanan

Jika cakupan kredensi tidak menentukan layanan yang sama dengan host header, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Credential should be scoped to correct service: 'service'
```

String penghentian

Jika cakupan kredensi tidak berakhir dengan aws4_request, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

Kesalahan penandatanganan kunci

Kesalahan yang disebabkan oleh derivasi kunci penandatanganan yang salah atau penggunaan kriptografi yang tidak tepat lebih sulit untuk dipecahkan. Setelah Anda memverifikasi bahwa string kanonik dan string yang akan ditandatangani sudah benar, Anda juga dapat memeriksa salah satu masalah berikut:

- Kunci akses rahasia tidak cocok dengan ID kunci akses yang Anda tentukan.
- Ada masalah dengan kode derivasi kunci Anda.

Untuk memverifikasi bahwa kunci rahasia cocok dengan ID kunci akses, Anda dapat mengujinya dengan implementasi kerja yang diketahui. Misalnya, gunakan AWS SDK atau AWS CLI untuk membuat permintaan AWS. Sebagai contoh, lihat Minta contoh tanda tangan

IAMJSONreferensi kebijakan

Bagian ini menyajikan sintaks rinci, deskripsi, dan contoh elemen, variabel, dan logika evaluasi JSON kebijakan dalam. IAM Untuk informasi umum selengkapnya, lihat Ikhtisar JSON kebijakan.

Referensi ini mencakup bagian berikut.

- IAMJSONreferensi elemen kebijakan — Pelajari lebih banyak tentang elemen yang dapat Anda gunakan ketika membuat kebijakan. Lihat contoh kebijakan tambahan dan pelajari tentang ketentuan, jenis data yang didukung, dan bagaimana digunakan dalam berbagai layanan.
- Logika evaluasi kebijakanBagian ini menjelaskan AWS permintaan, bagaimana mereka diautentikasi, dan bagaimana AWS menggunakan kebijakan untuk menentukan akses ke sumber daya.
- Tata bahasa IAM JSON kebijakan — Bagian ini menyajikan tata bahasa formal untuk bahasa yang digunakan untuk membuat kebijakan diIAM.
- AWS kebijakan terkelola untuk fungsi pekerjaan— Bagian ini mencantumkan semua AWS kebijakan terkelola yang secara langsung memetakan ke fungsi pekerjaan umum di industri TI. Gunakan kebijakan ini untuk memberikan izin yang diperlukan untuk melaksanakan tugas yang diharapkan dari seseorang dalam fungsi pekerjaan tertentu. Kebijakan ini mengonsolidasikan izin untuk banyak layanan ke dalam satu kebijakan.
- AWS kunci konteks kondisi global— Bagian ini mencakup daftar semua AWS kunci kondisi global yang dapat Anda gunakan untuk membatasi izin dalam IAM kebijakan.
- IAMdan kunci konteks AWS STS kondisi— Bagian ini mencakup daftar semua IAM dan AWS STS kunci kondisi yang dapat Anda gunakan untuk membatasi izin dalam IAM kebijakan.
- Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan - Bagian ini menyajikan daftar semua AWS APIoperasi yang dapat Anda gunakan sebagai izin dalam IAM kebijakan. Ini juga mencakup kunci kondisi khusus layanan yang dapat digunakan untuk lebih menyempurnakan permintaan.

IAMJSONreferensi elemen kebijakan

JSONdokumen kebijakan terdiri dari unsur-unsur. Elemen-elemen ini tercantum di sini dalam urutan umum Anda menggunakannya dalam kebijakan. Urutan elemen tidak penting—misalnya, elemen `Resource` dapat muncul sebelum elemen `Action`. Anda tidak diwajibkan untuk menentukan elemen `Condition` apa pun dalam kebijakan. Untuk mempelajari lebih lanjut tentang struktur umum dan tujuan dokumen JSON kebijakan, lihatIkhtisar JSON kebijakan.

Beberapa elemen JSON kebijakan saling eksklusif. Ini berarti bahwa Anda tidak dapat membuat kebijakan yang menggunakan keduanya. Misalnya, Anda tidak dapat menggunakan `Action` dan `NotAction` dalam pernyataan kebijakan yang sama. Pasangan lain yang saling eksklusif termasuk `Principal`/`NotPrincipal` dan `Resource`/`NotResource`.

Detail tentang apa saja yang termasuk ke dalam kebijakan berbeda-beda untuk setiap layanan, tergantung pada tindakan apa yang disediakan layanan, tipe sumber daya apa yang ada di dalamnya, dan sebagainya. Ketika Anda menulis kebijakan untuk layanan tertentu, akan sangat membantu jika Anda melihat contoh kebijakan untuk layanan tersebut. Untuk daftar semua layanan yang mendukungIAM, dan untuk tautan ke dokumentasi dalam layanan yang membahas IAM dan kebijakan tersebut, lihatAWS layanan yang bekerja dengan IAM.

Saat Anda membuat atau mengedit JSON kebijakan, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAMmengidentifikasi kesalahan JSON sintaks, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat Validasi kebijakan IAMAccess Analyzer.

Topik

- IAMJSONelemen kebijakan: Action
- IAMJSONelemen kebijakan: NotAction
- IAMJSONelemen kebijakan: Resource
- IAMJSONelemen kebijakan: NotResource
- IAMJSONelemen kebijakan: Condition
- Elemen kebijakan IAM: Variabel dan tanda
- Elemen kebijakan IAM JSON: Tipe data yang didukung

IAMJSONelemen kebijakan: Version

> **Catatan disambiguasi**
>
> Elemen `Version` JSON kebijakan ini berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan, di sisi lain, dibuat saat Anda membuat perubahan pada kebijakan yang dikelola pelanggan diIAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola. Jika Anda mencari informasi tentang dukungan beberapa versi yang tersedia untuk kebijakan terkelola, lihat the section called “Kebijakan IAM versioning”.

Elemen kebijakan `Version` menetapkan aturan sintaksis bahasa yang akan digunakan untuk memproses kebijakan. Untuk menggunakan semua fitur kebijakan yang tersedia, sertakan `Version` elemen berikut di luar `Statement` elemen di semua kebijakan Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

IAM mendukung nilai elemen `Version` berikut:

- `2012-10-17`. Ini adalah versi bahasa kebijakan saat ini, dan Anda harus selalu menyertakan elemen `Version` dan mengaturnya ke `2012-10-17`. Jika tidak, Anda tidak dapat menggunakan fitur seperti variabel kebijakan yang diperkenalkan dengan versi ini.
- `2008-10-17`. Ini adalah versi terdahulu dari bahasa kebijakan. Anda mungkin melihat versi ini pada kebijakan lama yang sudah ada. Jangan gunakan versi ini untuk kebijakan baru atau saat Anda memperbarui kebijakan yang sudah ada. Fitur yang lebih baru, seperti variabel kebijakan, tidak akan berfungsi dengan kebijakan Anda. Misalnya, variabel seperti `${aws:username}` tidak diakui sebagai variabel dan diperlakukan sebagai string literal di dalam kebijakan.

IAMJSONelemen kebijakan: Id

Elemen `Id` ini menentukan pengidentifikasi opsional untuk kebijakan. ID digunakan secara berbeda di layanan yang berbeda. ID diperbolehkan dalam kebijakan berbasis sumber daya, tetapi tidak dalam kebijakan berbasis identitas.

Untuk layanan yang memungkinkan Anda mengatur `ID` elemen, kami sarankan Anda menggunakan UUID (GUID) untuk nilai, atau memasukkan UUID sebagai bagian dari ID untuk memastikan keunikan.

```
{
  "Version": "2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Beberapa AWS layanan (misalnya, Amazon SQS atau AmazonSNS) mungkin memerlukan elemen ini dan memiliki persyaratan keunikan untuk itu. Untuk informasi khusus layanan tentang penulisan kebijakan, lihat dokumentasi untuk layanan yang sedang Anda kerjakan.

IAMJSONelemen kebijakan: Statement

Elemen `Statement` adalah elemen utama kebijakan ini. Elemen ini wajib diisi. Elemen `Statement` dapat berisi satu pernyataan atau serangkaian pernyataan individu. Setiap blok pernyataan individu harus ditutup dengan tanda kurung kurawal { }. Untuk beberapa pernyataan, larik harus dilampirkan dalam kurung persegi [ ].

```
"Statement": [{...},{...},{...}]
```

Contoh berikut menunjukkan kebijakan yang berisi susunan tiga pernyataan di dalam satu elemen `Statement` tunggal. (Kebijakan ini memungkinkan Anda mengakses “folder rumah” Anda sendiri di konsol Amazon S3.) Kebijakan ini mencakup variabel `aws:username`, yang diganti selama evaluasi kebijakan dengan nama pengguna dari permintaan. Untuk informasi selengkapnya, lihat Pengantar.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
```

```
    }
  ]
}
```

IAMJSONelemen kebijakan: Sid

Anda dapat memberikan `Sid` (ID pernyataan) sebagai pengenal opsional untuk pernyataan kebijakan. Anda dapat menetapkan nilai `Sid` untuk setiap pernyataan dalam rangkaian pernyataan. Anda dapat menggunakan `Sid` nilai sebagai deskripsi untuk pernyataan kebijakan. Dalam layanan yang memungkinkan Anda menentukan `ID` elemen, seperti SQS danSNS, `Sid` nilainya hanyalah sub-ID dari ID dokumen kebijakan. DalamIAM, `Sid` nilainya harus unik dalam suatu JSON kebijakan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

`Sid`Elemen ini mendukung ASCII huruf besar (A-Z), huruf kecil (a-z), dan angka (0-9).

IAMtidak mengekspos `Sid` di. IAM API Anda tidak dapat mengambil pernyataan tertentu berdasarkan ID ini.

> **Note**
>
> Beberapa AWS layanan (misalnya, Amazon SQS atau AmazonSNS) mungkin memerlukan elemen ini dan memiliki persyaratan keunikan untuk itu. Untuk informasi khusus layanan tentang kebijakan penulisan, lihat dokumentasi untuk layanan tempat Anda bekerja.

IAMJSONelemen kebijakan: Effect

Elemen `Effect` diperlukan dan menentukan apakah pernyataan tersebut mengakibatkan diizinkan atau ditolak. Nilai yang valid untuk `Effect` adalah `Allow` dan `Deny`. `Effect`Nilainya peka huruf besar/kecil.

```
"Effect":"Allow"
```

Secara default, akses ke sumber daya ditolak. Untuk mengizinkan akses ke sumber daya, Anda harus mengatur elemen `Effect` ke `Allow`. Untuk menimpa izinkan (misalnya, untuk menimpa izinkan yang berlaku sebaliknya), Anda mengatur elemen `Effect` ke `Deny`. Untuk informasi selengkapnya, lihat Logika evaluasi kebijakan.

AWS JSONelemen kebijakan: Principal

Gunakan `Principal` elemen dalam JSON kebijakan berbasis sumber daya untuk menentukan prinsipal yang diizinkan atau ditolak akses ke sumber daya.

Anda harus menggunakan `Principal` elemen dalam kebijakan berbasis sumber daya. Beberapa layanan mendukung kebijakan berbasis sumber daya, termasuk. IAM Jenis kebijakan IAM berbasis sumber daya adalah kebijakan kepercayaan peran. Dalam IAM peran, gunakan `Principal` elemen dalam kebijakan kepercayaan peran untuk menentukan siapa yang dapat mengambil peran tersebut. Untuk akses akun silang, Anda harus menentukan pengidentifikasi 12-digit dari akun tepercaya. Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? .

> **Note**
>
> Setelah membuat peran, Anda dapat mengubah akun menjadi “*” agar semua orang dapat mengambil peran tersebut. Jika Anda melakukannya, kami sangat menyarankan Anda untuk membatasi siapa yang dapat mengakses peran tersebut melalui cara lain, seperti elemen `Condition` yang membatasi akses ke alamat IP tertentu saja. Jangan biarkan peran Anda dapat diakses semua orang!

Contoh sumber daya lain yang mendukung kebijakan berbasis sumber daya termasuk bucket Amazon S3 atau AWS KMS key.

Anda tidak dapat menggunakan `Principal` elemen dalam kebijakan berbasis identitas. Kebijakan berbasis identitas adalah kebijakan izin yang Anda lampirkan ke IAM identitas (pengguna, grup, atau peran). Dalam kasus tersebut, kepala sekolah secara implisit adalah identitas di mana kebijakan dilampirkan.

Topik

Cara menentukan kepala sekolah

Anda menentukan prinsipal dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal.

Anda dapat menyebutkan salah satu prinsip dasar berikut dalam kebijakan:

- Akun AWS dan pengguna root
- IAMperan
- Sesi peran
- IAMpengguna
- Sesi pengguna federasi
- AWS layanan
- Semua kepala sekolah

Anda tidak dapat mengidentifikasi grup pengguna sebagai prinsipal dalam kebijakan (seperti kebijakan berbasis sumber daya) karena grup terkait dengan izin, bukan autentikasi, dan prinsipal adalah entitas yang diautentikasi. IAM

Anda dapat menentukan lebih dari satu prinsipal untuk masing-masing tipe prinsipal dalam bagian berikut menggunakan array. Susunan dapat mengambil satu atau beberapa nilai. Saat Anda

menentukan lebih dari satu prinsipal dalam suatu elemen, Anda memberikan izin kepada setiap prinsipal. Ini logis OR dan bukan logisAND, karena Anda mengautentikasi sebagai satu prinsipal pada satu waktu. Jika Anda menyertakan lebih dari satu nilai, gunakan tanda kurung siku ([dan]) dan comma-delimit setiap entri untuk array. Contoh kebijakan berikut mendefinisikan izin untuk akun 123456789012 atau akun 555555555555.

```
"Principal" : {
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> Anda tidak dapat menggunakan wildcard untuk mencocokkan bagian dari nama utama atauARN.

Akun AWS utama

Anda dapat menentukan Akun AWS pengidentifikasi dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Ini mendelegasikan wewenang ke akun. Ketika Anda mengizinkan akses ke akun yang berbeda, administrator di akun itu kemudian harus memberikan akses ke identitas (IAMpengguna atau peran) di akun itu. Saat Anda menentukan Akun AWS, Anda dapat menggunakan akun ARN (arn:aws:iam::*account-ID*:root), atau bentuk singkat yang terdiri dari awalan `"AWS":` diikuti dengan ID akun.

Misalnya, memberikan ID akun `123456789012`, Anda dapat menggunakan dari metode berikut untuk menyebutkan akun tersebut di elemen `Principal`:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

Akun ARN dan ID akun yang dipersingkat berperilaku dengan cara yang sama. Keduanya mendelegasikan izin ke akun. Menggunakan akun ARN dalam `Principal` elemen tidak membatasi izin hanya untuk pengguna root akun.

> **Note**
>
> Saat Anda menyimpan kebijakan berbasis sumber daya yang menyertakan ID akun yang dipersingkat, layanan dapat mengubahnya menjadi prinsipal. ARN Ini tidak mengubah fungsionalitas kebijakan.

Beberapa AWS layanan mendukung opsi tambahan untuk menentukan pokok akun. Misalnya, Amazon S3 memungkinkan Anda menentukan ID pengguna canonik menggunakan format berikut:

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

Anda juga dapat menentukan lebih dari satu Akun AWS, (atau ID pengguna kanonik) sebagai prinsipal menggunakan array. Misalnya, Anda dapat menentukan prinsipal dalam kebijakan bucket menggunakan ketiga metode tersebut.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
}
```

Prinsipal peran IAM

Anda dapat menentukan prinsip IAM peran ARNs dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. IAMPeran adalah identitas. DiIAM, identitas adalah sumber daya tempat Anda dapat menetapkan izin. Peran mempercayai identitas lain yang diautentikasi untuk mengambil peran itu. Ini termasuk prinsipal di AWS atau pengguna dari penyedia identitas eksternal (iDP). Ketika prinsipal atau identitas mengambil peran, mereka menerima kredensil keamanan sementara dengan izin peran yang diasumsikan. Ketika mereka menggunakan kredensi sesi tersebut untuk melakukan operasi di AWS, mereka menjadi kepala sesi peran.

IAMperan adalah identitas yang ada di dalamnyaIAM. Peran mempercayai identitas lain yang diautentikasi, seperti prinsipal di AWS atau pengguna dari penyedia identitas eksternal. Ketika kepala sekolah atau identitas mengambil peran, mereka menerima kredensi keamanan sementara. Mereka

kemudian dapat menggunakan kredensi tersebut sebagai prinsipal sesi peran untuk melakukan operasi di AWS.

Saat Anda menentukan prinsipal peran dalam kebijakan berbasis sumber daya, izin efektif untuk prinsipal dibatasi oleh jenis kebijakan apa pun yang membatasi izin untuk peran tersebut. Ini termasuk kebijakan sesi dan batas izin. Untuk informasi selengkapnya tentang bagaimana izin efektif untuk sesi peran dievaluasi, lihat. Logika evaluasi kebijakan

Untuk menentukan peran ARN dalam `Principal` elemen, gunakan format berikut:

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> Jika `Principal` elemen Anda dalam kebijakan kepercayaan peran berisi ARN yang menunjuk ke IAM peran tertentu, maka elemen tersebut akan ARN berubah menjadi ID utama unik peran saat Anda menyimpan kebijakan. Hal ini membantu memitigasi risiko seseorang meningkatkan hak istimewa mereka dengan menghapus dan membuat ulang peran. Anda biasanya tidak melihat ID ini di konsol, karena IAM menggunakan transformasi terbalik kembali ke peran ARN saat kebijakan kepercayaan ditampilkan. Namun, jika Anda menghapus peran, maka Anda memutuskan hubungan. Kebijakan tidak lagi berlaku, bahkan jika Anda membuat ulang peran tersebut karena peran baru memiliki ID utama baru yang tidak cocok dengan ID yang disimpan dalam kebijakan kepercayaan. Ketika ini terjadi, ID utama muncul dalam kebijakan berbasis sumber daya karena AWS tidak dapat lagi memetakannya kembali ke yang validARN. Hasil akhirnya adalah jika Anda menghapus dan membuat ulang peran yang direferensikan dalam `Principal` elemen kebijakan kepercayaan, Anda harus mengedit peran dalam kebijakan untuk mengganti ID utama dengan yang benar. ARN ARNSekali lagi berubah menjadi ID utama peran yang baru saat Anda menyimpan kebijakan.

Atau, Anda dapat menentukan prinsipal peran sebagai prinsipal dalam kebijakan berbasis sumber daya atau membuat kebijakan izin luas yang menggunakan kunci kondisi. `aws:PrincipalArn` Saat Anda menggunakan kunci ini, prinsipal sesi peran diberikan izin berdasarkan peran ARN yang diasumsikan, dan bukan sesi ARN yang dihasilkan. Karena AWS tidak mengonversi kunci kondisi ARNs menjadiIDs, izin yang diberikan ke peran ARN tetap ada jika Anda menghapus peran dan kemudian membuat peran baru dengan nama yang sama. Jenis kebijakan berbasis identitas, seperti batas izin atau kebijakan sesi, tidak membatasi izin yang diberikan menggunakan kunci

`aws:PrincipalArn` kondisi dengan wildcard (*) di `Principal` elemen, kecuali kebijakan berbasis identitas berisi penolakan eksplisit.

Kepala sesi peran

Anda dapat menentukan sesi peran dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Ketika prinsipal atau identitas mengambil peran, mereka menerima kredensil keamanan sementara dengan izin peran yang diasumsikan. Ketika mereka menggunakan kredensi sesi tersebut untuk melakukan operasi di AWS, mereka menjadi kepala sesi peran.

Format yang Anda gunakan untuk prinsipal sesi peran tergantung pada AWS STS operasi yang digunakan untuk mengambil peran.

Selain itu, administrator dapat merancang proses untuk mengontrol bagaimana sesi peran dikeluarkan. Misalnya, mereka dapat memberikan solusi satu-klik untuk penggunanya yang membuat nama sesi yang dapat diprediksi. Jika administrator melakukan ini, Anda dapat menggunakan prinsip sesi peran dalam kebijakan atau kunci kondisi. Jika tidak, Anda dapat menentukan peran ARN sebagai prinsipal dalam kunci `aws:PrincipalArn` kondisi. Cara Anda menentukan peran sebagai prinsipal dapat mengubah izin efektif untuk sesi yang dihasilkan. Untuk informasi selengkapnya, lihat Prinsipal peran IAM.

Prinsip sesi peran yang diasumsikan

Prinsipal sesi peran yang diasumsikan adalah prinsip sesi yang dihasilkan dari penggunaan AWS STS `AssumeRole`operasi. Untuk informasi lebih lanjut tentang prinsipal mana yang dapat mengambil peran menggunakan operasi ini, lihat. Bandingkan AWS STS kredensialnya

Untuk menentukan sesi peran yang diasumsikan ARN dalam `Principal` elemen, gunakan format berikut:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Saat Anda menentukan sesi peran yang diasumsikan dalam `Principal` elemen, Anda tidak dapat menggunakan wildcard “*” yang berarti semua sesi. Prinsipal harus selalu menamai sesi tertentu.

OIDCkepala sekolah sesi

Prinsipal OIDC sesi adalah prinsipal sesi yang dihasilkan dari penggunaan AWS STS `AssumeRoleWithWebIdentity`operasi. Anda dapat menggunakan OIDC penyedia eksternal

(iDP) untuk masuk, lalu mengambil IAM peran menggunakan operasi ini. Ini memanfaatkan federasi identitas dan mengeluarkan sesi peran. Untuk informasi lebih lanjut tentang prinsipal mana yang dapat mengambil peran menggunakan operasi ini, lihat. Bandingkan AWS STS kredensialnya

Ketika Anda mengeluarkan peran dari OIDC penyedia, Anda mendapatkan jenis prinsipal sesi khusus ini yang mencakup informasi tentang OIDC penyedia.

Gunakan tipe utama ini dalam kebijakan Anda untuk mengizinkan atau menolak akses berdasarkan penyedia identitas web tepercaya. Untuk menentukan sesi OIDC peran ARN dalam `Principal` elemen kebijakan kepercayaan peran, gunakan format berikut:

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

SAMLkepala sekolah sesi

Prinsipal SAML sesi adalah prinsipal sesi yang dihasilkan dari penggunaan AWS STS `AssumeRoleWithSAML`operasi. Anda dapat menggunakan penyedia SAML identitas eksternal (iDP) untuk masuk, lalu mengambil IAM peran menggunakan operasi ini. Ini memanfaatkan federasi identitas dan mengeluarkan sesi peran. Untuk informasi lebih lanjut tentang prinsipal mana yang dapat mengambil peran menggunakan operasi ini, lihat. Bandingkan AWS STS kredensialnya

Ketika Anda mengeluarkan peran dari penyedia SAML identitas, Anda mendapatkan jenis prinsipal sesi khusus ini yang mencakup informasi tentang penyedia SAML identitas.

Gunakan tipe utama ini dalam kebijakan Anda untuk mengizinkan atau menolak akses berdasarkan penyedia SAML identitas tepercaya. Untuk menentukan sesi peran SAML identitas ARN dalam `Principal` elemen kebijakan kepercayaan peran, gunakan format berikut:

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

IAMprinsipal pengguna

Anda dapat menentukan IAM pengguna dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal.

> **Note**
>
> Dalam `Principal` elemen, bagian nama pengguna dari Amazon Resource Name (ARN) peka huruf besar/kecil.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

Saat Anda menentukan pengguna di elemen `Principal`, Anda tidak dapat menggunakan wildcard (*) yang berarti “semua pengguna”. Prinsipal harus selalu memberi nama pengguna tertentu.

> **Important**
>
> Jika `Principal` elemen Anda dalam kebijakan kepercayaan peran berisi ARN yang menunjuk ke IAM pengguna tertentu, maka IAM ubah ARN menjadi ID utama unik pengguna saat Anda menyimpan kebijakan. Hal ini membantu memitigasi risiko seseorang meningkatkan hak istimewa mereka dengan menghapus dan membuat ulang pengguna. Anda biasanya tidak melihat ID ini di konsol, karena ada juga transformasi terbalik kembali ke pengguna ARN saat kebijakan kepercayaan ditampilkan. Namun, jika Anda menghapus pengguna, maka Anda memutuskan hubungan. Kebijakan tidak lagi berlaku, bahkan saat Anda membuat ulang pengguna. Itu karena pengguna baru memiliki ID utama baru yang tidak cocok dengan ID yang disimpan dalam kebijakan kepercayaan. Ketika ini terjadi, ID utama muncul dalam kebijakan berbasis sumber daya karena AWS tidak dapat lagi memetakannya kembali ke yang validARN. Hasilnya adalah jika Anda menghapus dan membuat ulang pengguna yang direferensikan dalam `Principal` elemen kebijakan kepercayaan, Anda harus mengedit peran untuk mengganti ID utama yang sekarang salah

dengan yang benar. ARN IAMsekali lagi berubah ARN menjadi ID utama baru pengguna saat Anda menyimpan kebijakan.

IAMPrinsipal Pusat Identitas

Di Pusat IAM Identitas, prinsipal dalam kebijakan berbasis sumber daya harus didefinisikan sebagai Akun AWS kepala sekolah. Untuk menentukan akses, rujuk ARN peran set izin di blok kondisi. Untuk detailnya, lihat Mereferensikan set izin dalam kebijakan sumber daya, AmazonEKS, dan AWS KMSdi Panduan Pengguna Pusat IAM Identitas.

AWS STS prinsip sesi pengguna federasi

Anda dapat menentukan sesi pengguna federasi dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal.

> **Important**
>
> AWS merekomendasikan agar Anda menggunakan AWS STS sesi pengguna federasi hanya bila diperlukan, seperti ketika akses pengguna root diperlukan. Sebagai gantinya, gunakan peran untuk mendelegasikan izin.

Sebuah AWS STS prinsipal sesi pengguna federasi adalah prinsip sesi yang dihasilkan dari penggunaan AWS STS `GetFederationToken`operasi. Dalam hal ini, AWS STS menggunakan federasi identitas sebagai metode untuk mendapatkan token akses sementara alih-alih menggunakan IAM peran.

Masuk AWS, IAM pengguna atau Pengguna root akun AWS dapat mengautentikasi menggunakan kunci akses jangka panjang. Untuk informasi lebih lanjut tentang kepala sekolah mana yang dapat berfederasi menggunakan operasi ini, lihat. Bandingkan AWS STS kredensialnya

- IAMpengguna federasi — IAM Pengguna federasi menggunakan `GetFederationToken` operasi yang menghasilkan prinsipal sesi pengguna federasi untuk pengguna tersebut. IAM
- Pengguna root federasi — Pengguna root federasi menggunakan `GetFederationToken` operasi yang menghasilkan prinsipal sesi pengguna federasi untuk pengguna root tersebut.

Ketika IAM pengguna atau pengguna root meminta kredensi sementara dari AWS STS menggunakan operasi ini, mereka memulai sesi pengguna federasi sementara. Sesi ini ARN didasarkan pada identitas asli yang difederasi.

Untuk menentukan sesi pengguna federasi ARN dalam `Principal` elemen, gunakan format berikut:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

AWS prinsipal layanan

Anda dapat menentukan AWS layanan dalam `Principal` elemen kebijakan berbasis sumber daya atau kunci kondisi yang mendukung prinsipal. Prinsipal layanan adalah pengenal untuk suatu layanan.

IAMPeran yang dapat diasumsikan oleh AWS Layanan disebut peran layanan. Peran layanan harus menyertakan kebijakan kepercayaan. Kebijakan kepercayaan adalah kebijakan berbasis sumber daya yang melekat pada peran yang menentukan prinsip mana yang dapat mengambil peran tersebut. Beberapa peran layanan telah menetapkan kebijakan kepercayaan. Namun, dalam beberapa kasus, Anda harus menentukan prinsip utama layanan dalam kebijakan kepercayaan. Prinsip layanan dalam suatu IAM kebijakan tidak bisa`"Service": "*"`.

> **Important**
>
> Pengidentifikasi untuk prinsipal layanan mencakup nama layanan, dan biasanya dalam format berikut:
> *service-name*`.amazonaws.com`

Prinsipal layanan ditentukan oleh layanan. Anda dapat menemukan prinsipal layanan untuk beberapa layanan dengan membuka AWS layanan yang bekerja dengan IAM, memeriksa apakah layanan memiliki Ya di kolom Peran yang dikaitkan dengan layanan, dan membuka tautan Ya untuk melihat dokumentasi peran yang dikaitkan dengan layanan untuk layanan tersebut. Temukan bagian Izin Peran Ditautkan Layanan untuk layanan tersebut dapat melihat prinsipal layanan.

Contoh berikut menunjukkan kebijakan yang dapat dilampirkan pada peran layanan. Kebijakan ini memungkinkan dua layanan, Amazon ECS dan Elastic Load Balancing, untuk mengambil peran tersebut. Layanan kemudian dapat melakukan tugas yang diberikan oleh kebijakan izin yang ditetapkan untuk peran tersebut (tidak ditampilkan). Untuk menetapkan beberapa prinsipal layanan, Anda tidak menentukan dua elemen `Service`; Anda hanya dapat memiliki satu. Sebagai gantinya,

Anda menggunakan serangkaian dari beberapa prinsipal layanan sebagai nilai elemen `Service` tunggal.

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

AWS prinsip layanan di Wilayah keikutsertaan

Anda dapat meluncurkan sumber daya di beberapa AWS Wilayah dan beberapa Wilayah yang harus Anda pilih. Untuk daftar lengkap Wilayah yang harus Anda pilih, lihat Mengelola AWS Daerah di Referensi Umum AWSpanduan.

Ketika sebuah AWS layanan di Wilayah keikutsertaan membuat permintaan dalam Wilayah yang sama, format nama utama layanan diidentifikasi sebagai versi non-regional dari nama utama layanan mereka:

service-name`.amazonaws.com`

Ketika sebuah AWS layanan di Wilayah keikutsertaan membuat permintaan lintas wilayah ke Wilayah lain, format nama utama layanan diidentifikasi sebagai versi regional dari nama utama layanan mereka:

service-name`.`*{region}*`.amazonaws.com`

Misalnya, Anda memiliki SNS topik Amazon yang terletak di Wilayah `ap-southeast-1` dan bucket Amazon S3 yang terletak di Wilayah keikutsertaan. `ap-east-1` Anda ingin mengonfigurasi notifikasi bucket S3 untuk mempublikasikan pesan ke SNS topik. Untuk mengizinkan layanan S3 memposting pesan ke SNS topik, Anda harus memberikan `sns:Publish` izin utama layanan S3 melalui kebijakan akses berbasis sumber daya dari topik tersebut.

Jika Anda menentukan versi non-regionalisasi dari prinsipal layanan S3`s3.amazonaws.com`, dalam kebijakan akses topik, `sns:Publish` permintaan dari bucket ke topik akan gagal. Contoh berikut menentukan prinsip layanan S3 non-regionalisasi dalam elemen `Principal` kebijakan kebijakan akses topik. SNS

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Karena bucket terletak di Wilayah keikutsertaan dan permintaan dibuat di luar Wilayah yang sama, prinsipal layanan S3 muncul sebagai nama utama layanan regional,. `s3.ap-east-1.amazonaws.com` Anda harus menggunakan nama utama layanan regional ketika AWS layanan di Wilayah keikutsertaan membuat permintaan ke Wilayah lain. Setelah Anda menentukan nama utama layanan regional, jika bucket membuat `sns:Publish` permintaan ke SNS topik yang terletak di Wilayah lain, permintaan akan berhasil. Contoh berikut menentukan prinsipal layanan S3 regional dalam elemen `Principal` kebijakan kebijakan akses topik. SNS

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Kebijakan sumber daya atau daftar izin berbasis prinsip layanan untuk permintaan Lintas wilayah dari Wilayah keikutsertaan ke Wilayah lain hanya akan berhasil jika Anda menentukan nama utama layanan regional.

> **Note**
>
> Untuk kebijakan kepercayaan IAM peran, sebaiknya gunakan nama utama layanan non-regional. IAMsumber daya bersifat global dan oleh karena itu peran yang sama dapat digunakan di Wilayah mana pun.

Semua kepala sekolah

Anda dapat menggunakan wildcard (*) untuk menentukan semua prinsipal dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Kebijakan berbasis sumber dayaizin pemberian dan kunci kondisi digunakan untuk membatasi kondisi pernyataan kebijakan.

> **Important**
>
> Kami sangat menyarankan agar Anda tidak menggunakan wildcard (*) dalam `Principal` elemen kebijakan berbasis sumber daya dengan `Allow` efek kecuali Anda berniat untuk memberikan akses publik atau anonim. Jika tidak, tentukan prinsip, layanan, atau AWS akun dalam `Principal` elemen dan kemudian membatasi akses lebih lanjut dalam `Condition` elemen. Hal ini terutama berlaku untuk kebijakan kepercayaan IAM peran, karena mereka memungkinkan prinsipal lain untuk menjadi prinsipal di akun Anda.

Untuk kebijakan berbasis sumber daya, menggunakan wildcard (*) dengan `Allow` efek memberikan akses ke semua pengguna, termasuk pengguna anonim (akses publik). Untuk IAM pengguna dan kepala peran dalam akun Anda, tidak ada izin lain yang diperlukan. Untuk prinsipal di akun lain, mereka juga harus memiliki izin berbasis identitas di akun mereka yang memungkinkan mereka mengakses sumber daya Anda. Ini disebut akses lintas akun.

Untuk pengguna anonim, elemen-elemen berikut ini setara:

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

Anda tidak dapat menggunakan wildcard untuk mencocokkan bagian dari nama utama atauARN.

Contoh berikut menunjukkan kebijakan berbasis sumber daya yang dapat digunakan sebagai ganti Menentukan dengan *NotPrincipalDeny* untuk secara eksplisit menolak semua prinsipal kecuali yang ditentukan dalam elemen. `Condition`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

Informasi lain

Untuk informasi selengkapnya, lihat berikut ini:

- Contoh kebijakan bucket di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon
- Contoh kebijakan untuk Amazon SNS di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon
- Contoh SQS kebijakan Amazon di Panduan Pengembang Layanan Antrian Sederhana Amazon
- Kebijakan-kebijakan utama dalam AWS Key Management Service Panduan Pengembang
- Pengidentifikasi akun di Referensi Umum AWS
- OIDCfederasi

AWS JSONelemen kebijakan: NotPrincipal

Anda dapat menggunakan `NotPrincipal` elemen untuk menolak akses ke semua prinsipal kecuali IAM pengguna, pengguna federasi, IAM peran,, AWS layanan Akun AWS, atau prinsip lain yang ditentukan dalam elemen. `NotPrincipal`

Anda dapat menggunakannya dalam kebijakan berbasis sumber daya untuk beberapa AWS layanan, termasuk titik akhir. VPC Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya. Anda tidak dapat menggunakan `NotPrincipal` elemen dalam kebijakan IAM berbasis identitas atau dalam kebijakan kepercayaan IAM peran.

`NotPrincipal`harus digunakan dengan`"Effect":"Deny"`. Menggunakannya `"Effect":"Allow"` dengan tidak didukung.

> **Important**
>
> Sangat sedikit skenario yang membutuhkan penggunaan`NotPrincipal`. Kami menyarankan Anda menjelajahi opsi otorisasi lain sebelum Anda memutuskan untuk menggunakannya`NotPrincipal`. Saat Anda menggunakan`NotPrincipal`, pemecahan masalah efek dari beberapa jenis kebijakan bisa jadi sulit. Sebaiknya gunakan kunci `aws:PrincipalArn` konteks dengan operator ARN kondisi sebagai gantinya. Untuk informasi selengkapnya, lihat Semua kepala sekolah.

Menentukan dengan **`NotPrincipalDeny`**

Saat Anda menggunakannya `NotPrincipalDeny`, Anda juga harus menentukan akun prinsipal ARN yang tidak ditolak. Jika tidak, kebijakan ini dapat menolak akses ke seluruh akun yang berisi prinsipal. Bergantung pada layanan yang Anda sertakan dalam kebijakan Anda, AWS dapat memvalidasi akun terlebih dahulu, lalu pengguna. Jika pengguna peran yang diasumsikan (seseorang yang menggunakan peran) sedang dievaluasi, AWS mungkin memvalidasi akun terlebih dahulu, kemudian peran, dan kemudian pengguna peran yang dianggap. Pengguna peran yang diasumsikan diidentifikasi oleh nama sesi peran yang ditentukan saat mereka mengasumsikan peran. Oleh karena itu, kami sangat menyarankan agar Anda secara eksplisit menyertakan akun ARN untuk pengguna, atau menyertakan peran ARN untuk dan ARN untuk akun yang berisi peran tersebut.

> **Important**
>
> Jangan gunakan pernyataan kebijakan berbasis sumber daya yang menyertakan elemen `NotPrincipal` kebijakan dengan `Deny` efek bagi IAM pengguna atau peran yang memiliki kebijakan batas izin yang dilampirkan. `NotPrincipal`Elemen dengan `Deny` efek akan selalu menolak IAM prinsip apa pun yang memiliki kebijakan batas izin yang dilampirkan, terlepas dari nilai yang ditentukan dalam elemen. `NotPrincipal` Hal ini menyebabkan beberapa IAM pengguna atau peran yang seharusnya memiliki akses ke sumber daya kehilangan akses. Sebaiknya ubah pernyataan kebijakan berbasis sumber daya Anda untuk menggunakan operator kondisi `ArnNotEquals`dengan kunci `aws:PrincipalArn`konteks untuk membatasi akses, bukan elemen. `NotPrincipal` Untuk informasi tentang batas izin, lihatBatas izin untuk entitas IAM.

> **Note**
>
> Sebagai praktik terbaik, Anda harus memasukkan ARNs untuk akun dalam kebijakan Anda. Beberapa layanan memerlukan akunARN, meskipun ini tidak diperlukan dalam semua kasus. Setiap kebijakan yang ada tanpa persyaratan ARN akan terus berfungsi, tetapi kebijakan baru yang mencakup layanan ini harus memenuhi persyaratan ini. IAMtidak melacak layanan ini, dan karena itu merekomendasikan agar Anda selalu menyertakan akunARN.

Contoh berikut menunjukkan cara menggunakan `NotPrincipal` dan `"Effect": "Deny"` dalam pernyataan kebijakan yang sama dengan efektif.

Example Contoh IAM pengguna di akun yang sama atau berbeda

Dalam contoh berikut, semua prinsipal kecuali pengguna bernama Bob di Akun AWS 444455556666 secara eksplisit ditolak akses ke sumber daya. Perhatikan bahwa sebagai praktik terbaik, `NotPrincipal` elemen berisi pengguna Bob dan Bob milik (`arn:aws:iam::444455556666:root`). ARN Akun AWS Jika `NotPrincipal` elemen hanya berisi BobARN, efek kebijakan mungkin secara eksplisit menolak akses ke Akun AWS yang berisi pengguna Bob. Dalam beberapa kasus, pengguna tidak dapat memiliki izin lebih banyak daripada akun induknya, jadi jika akun Bob secara eksplisit ditolak maka Bob mungkin tidak dapat mengakses sumber daya tersebut.

Contoh ini berfungsi sebagaimana dimaksud ketika merupakan bagian dari pernyataan kebijakan dalam kebijakan berbasis sumber daya yang dilampirkan ke sumber daya yang sama atau berbeda Akun AWS (bukan 444455556666). Contoh ini tidak memberikan akses kepada Bob, namun hanya menghapus Bob dari daftar prinsipal yang secara eksplisit ditolak. Agar Bob dapat mengakses sumber daya, pernyataan kebijakan lainnya harus secara eksplisit mengizinkan akses menggunakan `"Effect": "Allow"`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:iam::444455556666:user/Bob",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::BUCKETNAME",
            "arn:aws:s3:::BUCKETNAME/*"
        ]
    }]
}
```

Example Contoh IAM peran dalam akun yang sama atau berbeda

Dalam contoh berikut, semua prinsipal kecuali pengguna peran yang dianggap bernama Akun AWS 444455556666 secara eksplisit ditolak akses ke cross-account-audit-app sumber daya. Sebagai praktik terbaik, `NotPrincipal` elemen berisi pengguna peran yang ARN diasumsikan (), peran (cross-account-read-only-perancross-account-audit-app), dan peran yang dimiliki (Akun AWS

444455556666). Jika `NotPrincipal` elemen kehilangan peran, efek kebijakan mungkin secara eksplisit menolak akses ke peran tersebut. ARN Demikian pula, jika `NotPrincipal` elemen tersebut ARN kehilangan peran Akun AWS tersebut, efek dari kebijakan tersebut mungkin secara eksplisit menolak akses ke Akun AWS dan semua entitas dalam akun itu. Dalam beberapa kasus, pengguna peran yang dianggap tidak dapat memiliki izin lebih dari peran induknya, dan peran tidak dapat memiliki izin lebih dari induknya Akun AWS, jadi ketika peran atau akun secara eksplisit ditolak aksesnya, pengguna peran yang diasumsikan mungkin tidak dapat mengakses sumber daya.

Contoh ini berfungsi sebagaimana dimaksud ketika merupakan bagian dari pernyataan kebijakan dalam kebijakan berbasis sumber daya yang dilampirkan ke sumber daya yang berbeda Akun AWS (bukan 444455556666). Contoh ini dengan sendirinya tidak mengizinkan akses ke pengguna peran yang diasumsikan cross-account-audit-app, itu hanya menghilangkan cross-account-audit-app dari daftar kepala sekolah yang secara eksplisit ditolak. Untuk memberikan cross-account-audit-app akses ke sumber daya, pernyataan kebijakan lain harus secara eksplisit mengizinkan penggunaan akses. `"Effect": "Allow"`

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:sts::444455556666:assumed-role/cross-account-read-only-role/cross-
account-audit-app",
            "arn:aws:iam::444455556666:role/cross-account-read-only-role",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::Bucket_AccountAudit",
            "arn:aws:s3:::Bucket_AccountAudit/*"
        ]
    }]
}
```

Saat Anda menentukan sesi peran yang diasumsikan di elemen `NotPrincipal`, Anda tidak dapat menggunakan wildcard (*) yang berarti “semua sesi”. Prinsipal harus selalu menamai sesi tertentu.

IAMJSONelemen kebijakan: Action

Elemen `Action` menguraikan tindakan khusus atau tindakan yang akan diizinkan atau ditolak. Pernyataan harus mencakup elemen `Action` atau `NotAction`. Setiap AWS layanan memiliki

serangkaian tindakan sendiri yang menggambarkan tugas yang dapat Anda lakukan dengan layanan itu. Misalnya, daftar tindakan untuk Amazon S3 dapat ditemukan di Menentukan Izin dalam Kebijakan diPanduan Pengguna Layanan Penyimpanan Sederhana Amazon, daftar tindakan untuk Amazon EC2 dapat ditemukan di EC2APIReferensi Amazon, dan daftar tindakan untuk AWS Identity and Access Management dapat ditemukan di Referensi. IAM API Untuk menemukan daftar tindakan untuk layanan lain, lihat dokumentasi API referensi untuk layanan ini.

Anda menentukan nilai menggunakan namespace layanan sebagai prefiks tindakan (`iam`, `ec2`, `sqs`, `sns`, `s3`, dll.) diikuti dengan nama tindakan yang mengizinkan atau menolak. Nama harus sesuai dengan tindakan yang didukung oleh layanan. Prefiks dan nama tindakan tidak sensitif dengan huruf besar-kecil. Misalnya, `iam:ListAccessKeys` sama dengan `IAM:listaccesskeys`. Contoh berikut menunjukkan elemen `Action` untuk layanan yang berbeda.

SQSAksi Amazon

```
"Action": "sqs:SendMessage"
```

EC2Aksi Amazon

```
"Action": "ec2:StartInstances"
```

IAMaksi

```
"Action": "iam:ChangePassword"
```

Tindakan Amazon S3

```
"Action": "s3:GetObject"
```

Anda dapat menentukan beberapa nilai untuk elemen `Action`.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

Anda dapat menggunakan wildcard (*) untuk memberikan akses ke semua tindakan yang ditawarkan AWS produk tertentu. Misalnya, elemen `Action` berikut berlaku untuk semua tindakan S3.

```
"Action": "s3:*"
```

Anda juga dapat menggunakan wildcard (*) sebagai bagian dari nama tindakan. Misalnya, `Action` elemen berikut berlaku untuk semua IAM tindakan yang mencakup string`AccessKey`, termasuk`CreateAccessKey`,`DeleteAccessKey`,`ListAccessKeys`, dan`UpdateAccessKey`.

```
"Action": "iam:*AccessKey*"
```

Beberapa layanan memungkinkan Anda membatasi tindakan yang tersedia. Misalnya, Amazon SQS memungkinkan Anda menyediakan hanya sebagian dari semua kemungkinan SQS tindakan Amazon. Dalam hal ini, wildcard * tidak memungkinkan kontrol penuh atas antrean; itu hanya memungkinkan subset tindakan yang telah Anda bagikan. Untuk informasi lebih lanjut, lihat Memahami Izin di Panduan Pengembang Amazon Simple Queue Service.

IAMJSONelemen kebijakan: NotAction

`NotAction` adalah elemen kebijakan tingkat lanjut yang secara eksplisit cocok dengan semuanya kecuali daftar tindakan yang ditentukan. Menggunakan `NotAction` dapat menghasilkan kebijakan yang lebih singkat dengan mencantumkan hanya beberapa tindakan yang harus tidak cocok, daripada menyertakan daftar panjang tindakan yang sesuai. Tindakan yang `NotAction` ditentukan dalam tidak terpengaruh oleh `Allow` atau `Deny` efek dalam pernyataan kebijakan. Hal ini, pada gilirannya, berarti bahwa semua tindakan atau layanan yang berlaku yang tidak terdaftar diperbolehkan jika Anda menggunakan efek `Allow`. Sebagai tambahan, tindakan atau layanan yang tidak terdaftar tersebut ditolak jika Anda menggunakan efek `Deny`. Saat Anda menggunakan `NotAction` dengan elemen `Resource`, Anda memberikan cakupan untuk kebijakan. Ini adalah bagaimana AWS menentukan tindakan atau layanan mana yang berlaku. Untuk informasi selengkapnya, lihat kebijakan contoh berikut.

NotAction dengan Izinkan

Anda dapat menggunakan `NotAction` elemen dalam pernyataan dengan `"Effect": "Allow"` untuk menyediakan akses ke semua tindakan dalam AWS layanan, kecuali untuk tindakan yang ditentukan dalam`NotAction`. Anda dapat menggunakannya dengan elemen `Resource` untuk menyediakan cakupan untuk kebijakan, membatasi tindakan yang diizinkan untuk tindakan yang dapat dilakukan pada sumber daya tertentu.

Contoh berikut memungkinkan pengguna mengakses semua tindakan Amazon S3 yang dapat dilakukan pada sumber daya S3 kecuali untuk menghapus bucket. Ini tidak memungkinkan pengguna untuk menggunakan API operasi `ListAllMyBuckets` S3, karena tindakan itu memerlukan sumber daya “*”. Kebijakan ini juga tidak mengizinkan tindakan dalam layanan lain, karena tindakan layanan lainnya tidak berlaku untuk sumber daya S3.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

Terkadang, Anda mungkin ingin memungkinkan akses ke sejumlah besar tindakan. Dengan menggunakan elemen `NotAction` secara efektif membalik pernyataan, yang menghasilkan daftar tindakan yang lebih singkat. Misalnya, karena AWS memiliki begitu banyak layanan, Anda mungkin ingin membuat kebijakan yang memungkinkan pengguna melakukan semuanya kecuali IAM tindakan akses.

Contoh berikut memungkinkan pengguna untuk mengakses setiap tindakan di setiap AWS layanan kecuali untukIAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Hati-hati saat menggunakan elemen `NotAction` dan `"Effect": "Allow"` dalam pernyataan yang sama atau dalam pernyataan yang berbeda dalam kebijakan. `NotAction` cocok dengan semua layanan dan tindakan yang tidak tercantum atau berlaku secara eksplist untuk sumber daya yang ditentukan, dan dapat mengakibatkan pemberian izin kepada pengguna lebih dari yang Anda inginkan.

NotAction dengan mendustakan

Anda dapat menggunakan elemen `NotAction` dalam pernyataan dengan `"Effect": "Deny"` untuk menolak akses ke semua sumber daya yang tercantum kecuali untuk tindakan yang ditentukan dalam elemen `NotAction`. Kombinasi ini tidak memungkinkan item yang terdaftar, tetapi sebaliknya secara eksplisit menolak tindakan yang tidak tercantum. Anda tetap harus mengizinkan tindakan yang ingin Anda izinkan.

Contoh bersyarat berikut menolak akses ke IAM non-tindakan jika pengguna tidak masuk menggunakan. MFA Jika pengguna masuk denganMFA, maka `"Condition"` tes gagal dan `"Deny"` pernyataan akhir tidak berpengaruh. Perhatikan, bagaimanapun, bahwa ini tidak akan memberikan pengguna akses ke tindakan apa pun; itu hanya akan secara eksplisit menolak semua tindakan lain kecuali IAM tindakan.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

Untuk contoh kebijakan yang menolak akses ke tindakan di luar Wilayah tertentu, kecuali untuk tindakan dari layanan tertentu, lihat AWS: Menolak akses AWS berdasarkan Wilayah yang diminta.

IAMJSONelemen kebijakan: Resource

`Resource`Elemen dalam pernyataan IAM kebijakan mendefinisikan objek atau objek yang berlaku untuk pernyataan tersebut. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`.

Anda menentukan sumber daya menggunakan Amazon Resource Name (ARN). Format ARN tergantung pada Layanan AWS dan sumber daya spesifik yang Anda maksud. Meskipun ARN formatnya bervariasi, Anda selalu menggunakan ARN untuk mengidentifikasi sumber daya. Untuk informasi selengkapnya tentang formatARNs, lihatIAM ARNs. Untuk informasi tentang cara menentukan sumber daya, lihat dokumentasi untuk layanan yang ingin Anda tulis pernyataan.

> **Note**
>
> Beberapa Layanan AWS tidak memungkinkan Anda untuk menentukan tindakan untuk sumber daya individu. Dalam kasus ini, tindakan apa pun yang Anda cantumkan di `NotAction` elemen `Action` atau berlaku untuk semua sumber daya dalam layanan tersebut. Ketika ini terjadi, Anda menggunakan karakter wildcard (`*`) dalam `Resource` elemen.

Contoh berikut mengacu pada SQS antrian Amazon tertentu.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

Contoh berikut mengacu pada IAM pengguna bernama Bob dalam file Akun AWS.

> **Note**
>
> Dalam `Resource` elemen, nama IAM pengguna peka huruf besar/kecil.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

Menggunakan wildcard di sumber daya ARNs

Anda dapat menggunakan karakter wildcard (*dan?) dalam segmen individu ARN (bagian yang dipisahkan oleh titik dua) untuk mewakili:

- Setiap kombinasi karakter (*)
- Setiap karakter tunggal (?)

Anda dapat menggunakan beberapa * atau ? karakter di setiap segmen. Jika * wildcard adalah karakter terakhir dari ARN segmen sumber daya, itu dapat diperluas untuk mencocokkan di luar batas titik dua. Kami menyarankan Anda menggunakan wildcard (*dan?) dalam ARN segmen yang dipisahkan oleh titik dua.

> **Note**
>
> Anda tidak dapat menggunakan wildcard di segmen layanan yang mengidentifikasi produk. AWS Untuk informasi selengkapnya tentang ARN segmen, lihat Identifikasi AWS sumber daya dengan Nama Sumber Daya Amazon (ARNs)

Contoh berikut mengacu pada semua IAM pengguna yang jalurnya`/accounting`.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

Contoh berikut mengacu pada semua item dalam bucket Amazon S3 spesifik.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
```

Karakter asterisk (*) dapat diperluas untuk menggantikan segala sesuatu di dalam segmen, termasuk karakter seperti garis miring maju (/) yang mungkin tampak sebagai pembatas dalam namespace

layanan tertentu. Misalnya, pertimbangkan Amazon S3 berikut ARN sebagai logika ekspansi wildcard yang sama berlaku untuk semua layanan.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*/test/*"
```

Wildcard ARN berlaku untuk semua objek berikut di bucket, tidak hanya objek pertama yang terdaftar.

```
amzn-s3-demo-bucket/1/test/object.jpg
amzn-s3-demo-bucket/1/2/test/object.jpg
amzn-s3-demo-bucket/1/2/test/3/object.jpg
amzn-s3-demo-bucket/1/2/3/test/4/object.jpg
amzn-s3-demo-bucket/1///test///object.jpg
amzn-s3-demo-bucket/1/test/.jpg
amzn-s3-demo-bucket//test/object.jpg
amzn-s3-demo-bucket/1/test/
```

Pertimbangkan dua objek terakhir dalam daftar sebelumnya. Nama objek Amazon S3 dapat dimulai atau diakhiri dengan karakter pembatas garis miring () pembatas konvensional. / Sementara / bekerja sebagai pembatas, tidak ada signifikansi khusus ketika karakter ini digunakan dalam sumber daya. ARN Hal ini diperlakukan sama dengan karakter valid lainnya. Tidak ARN akan cocok dengan objek berikut:

```
amzn-s3-demo-bucket/1-test/object.jpg
amzn-s3-demo-bucket/test/object.jpg
amzn-s3-demo-bucket/1/2/test.jpg
```

Menentukan beberapa sumber daya

Anda dapat menentukan beberapa sumber daya dalam `Resource` elemen dengan menggunakan arrayARNs. Contoh berikut mengacu pada dua tabel DynamoDB.

```
"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

Menggunakan variabel kebijakan dalam sumber daya ARNs

Dalam `Resource` elemen, Anda dapat menggunakan variabel JSON kebijakan di bagian ARN yang mengidentifikasi sumber daya tertentu (yaitu, di bagian belakangARN). Misalnya, Anda dapat menggunakan kunci `{aws:username}` sebagai bagian dari sumber daya ARN untuk menunjukkan bahwa nama pengguna saat ini harus disertakan sebagai bagian dari nama sumber daya. Contoh berikut menunjukkan bagaimana Anda dapat menggunakan tombol `{aws:username}` di elemen `Resource`. Kebijakan ini mengizinkan akses ke tabel Amazon DynamoDB yang cocok dengan nama pengguna saat ini.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:account-id:table/${aws:username}"
  }
}
```

Untuk informasi selengkapnya tentang variabel JSON kebijakan, lihatElemen kebijakan IAM: Variabel dan tanda.

IAMJSONelemen kebijakan: NotResource

`NotResource` adalah elemen kebijakan tingkat lanjut yang secara eksplisit cocok dengan setiap sumber daya kecuali yang ditentukan. Menggunakan `NotResource` dapat menghasilkan kebijakan yang lebih pendek dengan mencantumkan hanya beberapa sumber yang seharusnya tidak sesuai, daripada menyertakan daftar panjang sumber daya yang sesuai. Ini khususnya berguna untuk kebijakan yang menerapkan satu layanan AWS .

Misalnya, bayangkan Anda memiliki grup bernama `HRPayroll`. Anggota tidak `HRPayroll` boleh mengakses sumber daya Amazon S3 apa pun kecuali `Payroll` folder di `HRBucket` bucket. Kebijakan berikut secara jelas menolak akses ke semua sumber daya Amazon S3 selain sumber daya yang tercantum. Namun demikian, harap diperhatikan bahwa kebijakan ini tidak memberikan akses ke sumber daya apa pun kepada pengguna.

```
{
  "Version": "2012-10-17",
  "Statement": {
```

```
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
    ]
  }
}
```

Biasanya, untuk secara eksplisit menolak akses ke sumber daya yang akan Anda tuliskan kebijakan yang menggunakan `"Effect":"Deny"` dan yang menyertakan elemen `Resource` yang mencantumkan setiap folder secara terpisah. Namun, dalam hal ini, setiap kali Anda menambahkan folder ke`HRBucket`, atau menambahkan sumber daya ke Amazon S3 yang tidak boleh diakses, Anda harus menambahkan namanya ke daftar di. `Resource` Jika Anda menggunakan elemen `NotResource` sebagai gantinya, akses pengguna secara otomatis ditolak ke folder baru kecuali Anda menambahkan nama folder untuk elemen `NotResource`.

Saat menggunakan `NotResource`, Anda harus ingat bahwa sumber daya yang ditentukan dalam elemen ini hanya tindakan yang terbatas. Hal ini, pada gilirannya, membatasi semua sumber daya yang akan berlaku untuk tindakan tersebut. Dalam contoh di atas, kebijakan hanya memengaruhi tindakan Amazon S3 dan karena itu hanya sumber daya Amazon S3. Jika tindakan tersebut juga menyertakan EC2 tindakan Amazon, maka kebijakan tersebut tidak akan menolak akses ke EC2 sumber daya apa pun. Untuk mempelajari tindakan dalam layanan yang memungkinkan menentukan sumber daya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan. ARN

NotResource dengan elemen lain

Jangan pernah menggunakan elemen `"Effect": "Allow"`, `"Action": "*"`, dan `"NotResource": "arn:aws:s3:::HRBucket"` secara bersama-sama. Pernyataan ini sangat berbahaya, karena memungkinkan semua tindakan masuk AWS pada semua sumber daya kecuali bucket `HRBucket` S3. Ini bahkan memungkinkan pengguna menambahkan kebijakan yang memungkinkan mereka mengakses `HRBucket`. Jangan lakukan ini.

Hati-hati saat menggunakan elemen `NotResource` dan `"Effect": "Allow"` dalam pernyataan yang sama atau dalam pernyataan yang berbeda dalam kebijakan. `NotResource` memungkinkan semua layanan dan sumber daya yang tidak tercantum secara eksplist, dan dapat mengakibatkan pemberian izin kepada pengguna lebih dari yang Anda inginkan. Menggunakan elemen `NotResource` dan `"Effect": "Deny"` dalam pernyataan yang sama menolak layanan dan sumber daya yang secara tidak eksplisit tercantum.

IAMJSONelemen kebijakan: Condition

Elemen `Condition` (atau blok `Condition`) memungkinkan Anda menentukan ketentuan saat kebijiakan berlaku. Elemen `Condition` bersifat opsional. Dalam `Condition` elemen, Anda membangun ekspresi di mana Anda menggunakan operator kondisi (sama, kurang dari, dan lainnya) untuk mencocokkan kunci konteks dan nilai dalam kebijakan terhadap kunci dan nilai dalam konteks permintaan. Untuk mempelajari selengkapnya tentang konteks permintaan, lihat Komponen permintaan.

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

Kunci konteks yang Anda tentukan dalam kondisi kebijakan dapat berupa kunci konteks kondisi global atau kunci konteks khusus layanan. Kunci konteks kondisi global memiliki `aws:` awalan. Kunci konteks khusus layanan memiliki awalan layanan. Misalnya, Amazon EC2 memungkinkan Anda menulis kondisi menggunakan kunci `ec2:InstanceType` konteks, yang unik untuk layanan tersebut. Untuk melihat kunci IAM konteks khusus layanan dengan `iam:` awalan, lihat. IAMdan kunci konteks AWS STS kondisi

Nama kunci konteks tidak peka huruf besar/kecil. Misalnya, menyertakan kunci `aws:SourceIP` konteks setara dengan pengujian untuk`AWS:SourceIp`. Sensitivitas huruf besar/huruf dari nilai kunci konteks tergantung pada operator kondisi yang Anda gunakan. Misalnya, kondisi berikut mencakup `StringEquals` operator untuk memastikan bahwa hanya permintaan yang dibuat berdasarkan `johndoe` kecocokan. Nama pengguna `JohnDoe` ditolak aksesnya.

```
"Condition" : { "StringEquals" : { "aws:username" : "johndoe" }}
```

Ketentuan berikut menggunakan operator `StringEqualsIgnoreCase` untuk mencocokkan nama pengguna `johndoe` atau `JohnDoe`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "johndoe" }}
```

Beberapa kunci konteks mendukung pasangan kunci-nilai yang memungkinkan Anda menentukan bagian dari nama kunci. Contohnya termasuk kunci `aws:RequestTag/tag-key` konteks, kunci konteks AWS KMS `kms:EncryptionContext:encryption_context_key`, dan kunci `ResourceTag/tag-key` konteks yang didukung oleh beberapa layanan.

- Jika Anda menggunakan kunci `ResourceTag/tag-key` konteks untuk layanan seperti Amazon EC2, maka Anda harus menentukan nama kunci untuk`tag-key`.

- Nama kunci tidak peka huruf besar/kecil. Ini berarti jika Anda menentukan `"aws:ResourceTag/TagKey1": "Value1"` dalam elemen ketentuan kebijakan Anda, kemudian ketentuan tersebut cocok dengan kunci tanda sumber daya bernama `TagKey1` atau `tagkey1`, tetapi tidak keduanya.
- AWS layanan yang mendukung atribut ini memungkinkan Anda membuat beberapa nama kunci yang hanya berbeda berdasarkan kasus. Misalnya, Anda dapat menandai EC2 instance Amazon dengan `ec2=test1` dan`EC2=test2`. Saat Anda menggunakan kondisi seperti `"aws:ResourceTag/EC2": "test1"` untuk memungkinkan akses ke sumber daya tersebut, nama kunci cocok dengan kedua tanda, tetapi hanya satu nilai yang cocok. Hal ini dapat mengakibatkan kegagalan ketentuan yang tidak terduga.

> **Important**
>
> Sebagai praktik terbaik, pastikan bahwa anggota akun Anda mengikuti konvensi pemberian nama yang konsisten ketika memberi nama atribut pasangan nilai-kunci. Contohnya mencakup tanda atau konteks enkripsi AWS KMS . Anda dapat menerapkan ini menggunakan kunci `aws:TagKeys`konteks untuk penandaan, atau `kms:EncryptionContextKeys`untuk konteks AWS KMS enkripsi.

- Untuk daftar semua operator kondisi dan deskripsi tentang cara kerjanya, lihat Operator kondisi.
- Kecuali ditentukan lain, semua kunci konteks dapat memiliki beberapa nilai. Untuk deskripsi tentang cara menangani kunci konteks yang memiliki beberapa nilai, lihatKunci konteks multivaluasi.
- Untuk daftar semua kunci konteks yang tersedia secara global, lihatAWS kunci konteks kondisi global.
- Untuk kunci konteks kondisi yang ditentukan oleh setiap layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Konteks permintaan

Ketika kepala sekolah membuat permintaan AWS, AWS mengumpulkan informasi permintaan ke dalam konteks permintaan. Informasi ini digunakan untuk mengevaluasi dan mengotorisasi permintaan. Anda dapat menggunakan `Condition` elemen JSON kebijakan untuk menguji kunci konteks tertentu terhadap konteks permintaan. Misalnya, Anda dapat membuat kebijakan yang menggunakan kunci CurrentTime konteks aws: untuk memungkinkan pengguna melakukan tindakan hanya dalam rentang tanggal tertentu.

Ketika permintaan dikirimkan, AWS mengevaluasi setiap kunci konteks dalam kebijakan dan mengembalikan nilai true, false, not present, dan kadang-kadang null (string data kosong). Kunci konteks yang tidak ada dalam permintaan dianggap sebagai ketidakcocokan. Misalnya, kebijakan berikut memungkinkan penghapusan perangkat autentikasi multi-faktor (MFA) Anda sendiri, tetapi hanya jika Anda telah login menggunakan MFA dalam satu jam terakhir (3.600 detik).

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

Konteks permintaan dapat mengembalikan nilai-nilai berikut:

- Benar - Jika pemohon masuk menggunakan MFA dalam satu jam terakhir atau kurang, maka kondisi mengembalikan nilai true.
- False - Jika pemohon masuk menggunakan MFA lebih dari satu jam yang lalu, maka kondisi mengembalikan false.
- Tidak hadir — Jika pemohon membuat permintaan menggunakan kunci akses IAM pengguna mereka di AWS CLI atau AWS API, kunci tidak ada. Dalam hal ini, kuncinya tidak ada, dan kunci tidak akan cocok.
- Null — Untuk kunci konteks yang ditentukan oleh pengguna, seperti tag yang diteruskan dalam permintaan, dimungkinkan untuk menyertakan string kosong. Dalam hal ini, nilai dalam konteks permintaan adalah null. Nilai null mungkin akan mengembalikan nilai tersebut dalam beberapa kasus. Misalnya, jika Anda menggunakan operator *ForAllValues* kondisi multivalued dengan kunci *aws:TagKeys* konteks, Anda dapat mengalami hasil yang tidak terduga jika konteks permintaan mengembalikan null. Untuk informasi lebih lanjut, lihat aws: TagKeys andKunci konteks multivaluasi.

Blok ketentuan

Contoh berikut menunjukkan format dasar dari elemen `Condition`:

```
"Condition": {"StringLike": {"s3:prefix": ["janedoe/*"]}}
```

Nilai dari permintaan diwakili oleh kunci konteks, dalam hal ini`s3:prefix`. Nilai kunci konteks dibandingkan dengan nilai yang Anda tentukan sebagai nilai literal, seperti`janedoe/*`. Jenis perbandingan yang akan dibuat ditentukan oleh operator kondisi (di sini,`StringLike`). Anda dapat membuat ketentuan yang membandingkan string, tanggal, angka, dan lainnya menggunakan perbandingan umum Boolean seperti sama, lebih besar dari, dan kurang dari. Bila Anda menggunakan operator string atau ARNoperator, Anda juga dapat menggunakan variabel kebijakan dalam nilai kunci konteks. Contoh berikut termasuk `aws:username` variabel.

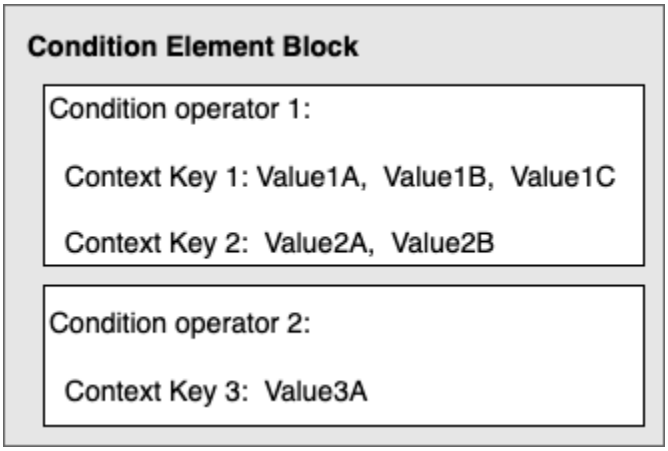
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

Dalam beberapa keadaan, kunci konteks dapat berisi beberapa nilai. Misalnya, permintaan ke Amazon DynamoDB mungkin meminta untuk mengembalikan atau memperbarui beberapa atribut dari tabel. Kebijakan untuk akses ke tabel DynamoDB dapat menyertakan `dynamodb:Attributes` kunci konteks, yang berisi semua atribut yang tercantum dalam permintaan. Anda dapat menguji beberapa atribut dalam daftar permintaan terhadap daftar atribut yang diizinkan di kebijakan dengan menggunakan operator kumpulan di elemen `Condition`. Untuk informasi selengkapnya, lihat Kunci konteks multivaluasi.

Ketika kebijakan dievaluasi selama permintaan, AWS ganti kunci dengan nilai yang sesuai dari permintaan. (Dalam contoh ini, AWS akan menggunakan tanggal dan waktu permintaan.) Kondisi dievaluasi untuk mengembalikan benar atau salah, yang kemudian diperhitungkan dalam apakah kebijakan tersebut secara keseluruhan mengizinkan atau menolak permintaan.

Beberapa nilai dalam suatu ketentuan

Sebuah `Condition` elemen dapat berisi beberapa operator kondisi, dan setiap operator kondisi dapat berisi beberapa pasangan kunci-nilai konteks. Gambar berikut mengilustrasikan hal ini.



Untuk informasi selengkapnya, lihat Kunci konteks multivaluasi.

IAMJSONelemen kebijakan: Operator kondisi

Gunakan operator ketentuan di elemen `Condition` untuk mencocokkan kunci kondisi dan nilai di dalam kebijakan terhadap nilai di dalam konteks permintaan. Untuk informasi selengkapnya tentang elemen `Condition`, lihat IAMJSONelemen kebijakan: Condition.

Operator ketentuan yang dapat Anda gunakan dalam kebijakan tergantung pada kunci kondisi yang Anda pilih. Anda dapat memilih kunci kondisi global atau kunci kondisi khusus layanan. Untuk mempelajari operator ketentuan mana yang dapat Anda gunakan pada kunci kondisi global, lihat AWS kunci konteks kondisi global. Untuk mempelajari operator kondisi mana yang dapat Anda gunakan untuk kunci kondisi khusus layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan yang ingin Anda lihat.

> **Important**
>
> Jika kunci yang Anda tentukan dalam kondisi kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok dan kondisinya salah. Jika kondisi kebijakan mengharuskan kunci tidak cocok, seperti `StringNotLike` atau`ArnNotLike`, dan kunci kanan tidak ada, kondisi tersebut benar. Logika ini berlaku untuk semua operator kondisi kecuali... IfExistsdan cek Null. Operator ini menguji apakah kuncinya ada (exists) dalam konteks permintaan.

Operator ketentuan dapat dikelompokkan ke dalam kategori berikut:

- Tali

- Numerik
- Tanggal dan waktu
- Boolean
- Biner
- Alamat IP
- Nama Sumber Daya Amazon (ARN) (tersedia hanya untuk beberapa layanan.)
- ... IfExists(memeriksa apakah nilai kunci ada sebagai bagian dari pemeriksaan lain)
- Pemeriksaan null (memeriksa apakah nilai kunci ada sebagai cek mandiri)

Operator ketentuan string

Operator ketentuan string memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke nilai string.

Operator ketentuan	Deskripsi
`StringEquals`	Kecocokan yang tepat, peka terhadap huruf besar-kecil
`StringNotEquals`	Pencocokan dinegasikan
`StringEqualsIgnoreCase`	Pencocokan yang tepat, mengabaikan huruf besar-kecil
`StringNotEqualsIgnoreCase`	Pencocokan yang dinegasikan, mengabaikan huruf besar-kecil
`StringLike`	Kecocokan kepekaan huruf besar-kecil. Nilainya dapat mencakup wildcard pencocokan multi-karakter (*) dan wildcard pencocokan karakter tunggal (?) di mana saja di string. Anda harus menentukan wildcard untuk mencapai kecocokan string paral. **Note** Jika kunci berisi beberapa nilai, `StringLike` dapat dikualifikasikan dengan operator yang ditetapkan—

Operator ketentuan	Deskripsi
	`ForAllValues:StringLike` dan `ForAnyValue:StringLike`. Untuk informasi selengkapnya, lihat Kunci konteks multivaluasi.
`StringNotLike`	Kecocokan dinegasikan kepekaan huruf besar-kecil. Nilai dapat mencakup wildcard pencocokan multi-karakter (*) atau wildcard pencocokan karakter tunggal (?) di mana saja di string.

Misalnya, pernyataan berikut berisi `Condition` elemen yang menggunakan `aws:PrincipalTag`kunci untuk menentukan bahwa prinsipal yang membuat permintaan harus ditandai dengan kategori `iamuser-admin` pekerjaan.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/job-category": "iamuser-admin"}}
  }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Dalam contoh ini, `aws:PrincipalTag/job-category` kunci hadir dalam konteks permintaan jika prinsipal menggunakan IAM pengguna dengan tag terlampir. Hal ini juga termasuk untuk prinsipal menggunakan IAM peran dengan tag terlampir atau tag sesi. Jika pengguna tanpa tanda mencoba melihat atau mengedit access key, ketentuan akan kembali `false` dan permintaan tersebut ditolak secara tersirat oleh pernyataan ini.

Anda dapat menggunakan variabel kebijakan dengan operator ketentuan `String`.

Contoh berikut menggunakan operator `StringLike` kondisi untuk melakukan pencocokan string dengan variabel kebijakan guna membuat kebijakan yang memungkinkan IAM pengguna menggunakan konsol Amazon S3 untuk mengelola “direktori home” miliknya sendiri di bucket Amazon S3. Kebijakan tersebut mengizinkan tindakan tertentu pada bucket S3 selama `s3:prefix` cocok dengan salah satu pola yang ditentukan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

Untuk contoh kebijakan yang menunjukkan cara menggunakan `Condition` elemen untuk membatasi akses ke sumber daya berdasarkan ID aplikasi dan ID pengguna untuk OIDC federasi, lihatAmazon S3: Memungkinkan pengguna Amazon Cognito mengakses objek di bucket mereka.

Pencocokan wildcard

Operator kondisi string melakukan pencocokan tanpa pola yang tidak menerapkan format yang telah ditentukan sebelumnya. ARNdan operator kondisi tanggal adalah bagian dari operator string yang menegakkan struktur pada nilai kunci kondisi. Saat Anda menggunakan StringLike atau StringNotLike

operator untuk kecocokan string paral dari tanggal ARN atau, pencocokan mengabaikan bagian struktur mana yang dikartu liar.

Misalnya, kondisi berikut mencari kecocokan sebagian dari ARN menggunakan operator kondisi yang berbeda.

Ketika ArnLike digunakan, partisi, layanan, account-id, resource-type, dan sebagian resource-id bagian ARN harus memiliki pencocokan yang tepat dengan dalam konteks permintaan. ARN Hanya wilayah dan jalur sumber daya yang memungkinkan pencocokan sebagian.

```
"Condition": {"ArnLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

Ketika StringLike digunakan sebagai pengganti ArnLike, pencocokan mengabaikan ARN struktur dan memungkinkan pencocokan paral, terlepas dari bagian yang dikartu liar.

```
"Condition": {"StringLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

ARN	ArnLike	StringLike
arn:aws:cloudtrail:us-west-2:111122223333:trail/keuangan	Pertandingan	Pertandingan
arn:aws:cloudtrail:us-east-2:111122223333:trail/keuangan/arsip	Pertandingan	Pertandingan
arn:aws:cloudtrail:us-east-2:444455556666:user /111122223333:trail/finance	Tidak ada kecocokan	Pertandingan

Operator ketentuan numerik

Operator ketentuan numerik memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke nilai bilangan bulat atau desimal.

Operator ketentuan	Deskripsi
`NumericEquals`	Pencocokan
`NumericNotEquals`	Pencocokan dinegasikan
`NumericLessThan`	Pencocokan “Kurang dari”
`NumericLessThanEqu als`	Pencocokan “kurang dari atau sama dengan”
`NumericGreaterThan`	Pencocokan “lebih besar dari”
`NumericGreaterThan Equals`	Pencocokan “lebih besar dari atau sama dengan”

Misalnya, pernyataan berikut memuat elemen `Condition` yang menggunakan operator ketentuan `NumericLessThanEquals` dengan kunci `s3:max-keys` untuk menentukan bahwa pemohon dapat membuat daftar hingga 10 objek dalam `amzn-s3-demo-bucket` sekali waktu.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Dalam contoh ini, kunci `s3:max-keys` selalu hadir dalam permintaan saat Anda melakukan operasi `ListBucket`. Jika kebijakan ini mengizinkan semua operasi Amazon S3 maka hanya operasi yang mencakup kunci konteks `max-keys` yang bernilai kurang dari atau sama dengan 10 akan diperbolehkan.

Anda tidak dapat menggunakan variabel kebijakan dengan operator ketentuan `Numeric`.

Operator ketentuan tanggal

Operator ketentuan tanggal memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke nilai tanggal/waktu. Anda menggunakan operator kondisi ini dengan `aws:CurrentTime`kunci atau `aws:EpochTime`kunci. Anda harus menentukan nilai tanggal/waktu dengan salah satu implementasi W3C dari format tanggal ISO 8601 atau dalam waktu epoch (). UNIX

> **Note**
>
> Wildcard tidak diizinkan untuk operator ketentuan tanggal.

Operator ketentuan	Deskripsi
`DateEquals`	Mencocokkan tanggal tertentu
`DateNotEquals`	Pencocokan dinegasikan
`DateLessThan`	Sesuai sebelum tanggal dan waktu tertentu
`DateLessThanEquals`	Sesuai pada atau sebelum tanggal dan waktu tertentu
`DateGreaterThan`	Sesuai setelah tanggal dan waktu tertentu
`DateGreaterThanEqu als`	Sesuai pada atau setelah tanggal dan waktu tertentu

Misalnya, pernyataan berikut memuat elemen `Condition` yang menggunakan operator ketentuan `DateGreaterThan` dengan kunci `aws:TokenIssueTime`. Ketentuan ini menyebutkan bahwa kredensial keamanan sementara yang digunakan untuk membuat permintaan tersebut diterbitkan pada 2020. Kebijakan ini dapat secara terprogram diperbarui setiap hari untuk memastikan anggota akun menggunakan kredensial baru.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"DateGreaterThan": {"aws:TokenIssueTime": "2020-01-01T00:00:01Z"}}
  }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Kunci `aws:TokenIssueTime` tersedia dalam konteks permintaan hanya ketika prinsipal menggunakan kredensial sementara untuk membuat permintaan. Kuncinya tidak ada di AWS CLI, AWS API, atau AWS SDKpermintaan yang dibuat menggunakan kunci akses. Dalam contoh ini, jika IAM pengguna mencoba untuk melihat atau mengedit kunci akses, permintaan ditolak.

Anda tidak dapat menggunakan variabel kebijakan dengan operator ketentuan `Date`.

Operator ketentuan Boolean

Ketentuan Boolean memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci "benar" atau "salah."

Operator ketentuan	Deskripsi
`Bool`	Pencocokan Boolean

Misalnya, kebijakan berbasis identitas ini menggunakan operator `Bool` kondisi dengan `aws:SecureTransport`kunci untuk menolak mereplikasi objek dan tag objek ke bucket tujuan dan isinya jika permintaan belum selesai. SSL

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
```

```
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Konteks `aws:SecureTransport` permintaan mengembalikan true atau false.

Anda dapat menggunakan variabel kebijakan dengan operator ketentuan `Boolean`.

Operator ketentuan biner

Operator ketentuan `BinaryEquals` memungkinkan Anda membangun elemen `Condition` yang menguji nilai kunci yang berada dalam format biner. Ini membandingkan nilai dari byte kunci yang ditentukan untuk byte terhadap perwakilan terkode base-64 dari nilai biner di dalam kebijakan.

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok.

Anda tidak dapat menggunakan variabel kebijakan dengan operator ketentuan `Binary`.

Operator ketentuan alamat IP

Operator kondisi alamat IP memungkinkan Anda membangun `Condition` elemen yang membatasi akses berdasarkan membandingkan kunci dengan IPv4 atau IPv6 alamat atau rentang alamat IP.

Anda menggunakan ini dengan kunci aws:SourceIp. Nilai harus dalam CIDR format standar (misalnya, 203.0.113.0/24 atau 2001:: 1234:5678: DB8 :/64). Jika Anda menentukan alamat IP tanpa awalan perutean terkait, IAM gunakan nilai awalan default dari. /32

Beberapa AWS dukungan layananIPv6, menggunakan:: untuk mewakili rentang 0s. Untuk mempelajari apakah suatu layanan mendukungIPv6, lihat dokumentasi untuk layanan tersebut.

Operator ketentuan	Deskripsi
IpAddress	Alamat IP atau rentang yang ditentukan
NotIpAddress	Semua alamat IP kecuali alamat dan rentang IP yang ditentukan

Misalnya, pernyataan berikut menggunakan operator ketentuan IpAddress dengan kunci aws:SourceIp untuk menentukan bahwa permintaan harus datang dari rentang IP 203.0.113.0 hingga 203.0.113.255.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"IpAddress": {"aws:SourceIp": "203.0.113.0/24"}}
  }
}
```

Kunci kondisi aws:SourceIp menyelesaikan ke alamat IP yang berasal dari permintaan asli. Jika permintaan berasal dari EC2 instans Amazon, aws:SourceIp evaluasi ke alamat IP publik instans.

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. aws:SourceIpKuncinya selalu ada dalam konteks permintaan, kecuali ketika pemohon menggunakan VPC titik akhir untuk membuat permintaan. Dalam hal ini, ketentuan mengembalikan false dan permintaan tersebut ditolak secara tersirat oleh pernyataan ini.

Anda tidak dapat menggunakan variabel kebijakan dengan operator ketentuan IpAddress.

Contoh berikut menunjukkan cara mencampur IPv4 dan IPv6 alamat untuk mencakup semua alamat IP valid organisasi Anda. Kami menyarankan agar Anda memperbarui kebijakan organisasi Anda

dengan rentang IPv6 alamat Anda selain IPv4 rentang yang sudah Anda miliki untuk memastikan kebijakan terus berfungsi saat Anda melakukan transisiIPv6.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
        ]
      }
    }
  }
}
```

Kunci `aws:SourceIp` kondisi hanya berfungsi dalam JSON kebijakan jika Anda memanggil yang diuji API secara langsung sebagai pengguna. Jika Anda menggunakan layanan untuk menelepon layanan target atas nama Anda, layanan target akan melihat alamat IP layanan panggilan, bukan alamat IP pengguna asal. Hal ini dapat terjadi, misalnya, jika Anda menggunakan AWS CloudFormation untuk memanggil Amazon EC2 untuk membuat instance untuk Anda. Saat ini tidak ada cara untuk meneruskan alamat IP asal melalui layanan panggilan ke layanan target untuk dievaluasi dalam JSON kebijakan. Untuk jenis API panggilan layanan ini, jangan gunakan tombol `aws:SourceIp` kondisi.

Operator kondisi Nama Sumber Daya Amazon (ARN)

Operator kondisi Amazon Resource Name (ARN) memungkinkan Anda membuat `Condition` elemen yang membatasi akses berdasarkan membandingkan kunci ke file. ARN ARNItu dianggap sebagai string.

Operator ketentuan	Deskripsi
`ArnEquals` , `ArnLike`	Pencocokan peka huruf besar/kecil dari. ARN Masing-masing dari enam komponen yang dibatasi titik dua dicentang secara terpisah dan masing-masing dapat menyertakan wildcard pencocokan multi-kar

Operator ketentuan	Deskripsi
	akter (*) atau wildcard pencocokan karakter tunggal (?). ARN Operator `ArnEquals` dan `ArnLike` kondisi berperilaku identik.
`ArnNotEquals` , `ArnNotLike`	Pencocokan yang dinegasikan untukARN. Operator `ArnNotEquals` dan `ArnNotLike` kondisi berperilaku identik.

Anda dapat menggunakan variabel kebijakan dengan operator ketentuan `ARN`.

Contoh kebijakan berbasis sumber daya berikut menunjukkan kebijakan yang dilampirkan ke SQS antrean Amazon yang ingin Anda kirimi pesan. SNS Ini memberi Amazon SNS izin untuk mengirim pesan ke antrian (atau antrian) pilihan Anda, tetapi hanya jika layanan mengirim pesan atas nama SNS topik Amazon tertentu (atau topik). Anda menentukan antrian di `Resource` bidang, dan SNS topik Amazon sebagai nilai untuk `SourceArn` kunci.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "123456789012"},
    "Action": "SQS:SendMessage",
    "Resource": "arn:aws:sqs:REGION:123456789012:QUEUE-ID",
    "Condition": {"ArnEquals": {"aws:SourceArn":
 "arn:aws:sns:REGION:123456789012:TOPIC-ID"}}
  }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Kunci `aws:SourceArn` muncul dalam konteks permintaan hanya jika sumber daya memicu layanan untuk menghubungi layanan lain atas nama pemilik sumber daya. Jika IAM pengguna mencoba untuk melakukan operasi ini secara langsung, kondisi kembali `false` dan permintaan secara implisit ditolak oleh pernyataan ini.

... IfExists operator kondisi

Anda dapat menambahkan `IfExists` ke akhir nama operator kondisi apa pun kecuali `Null` kondisi —misalnya,. `StringLikeIfExists` Anda melakukan ini untuk mengatakan “Jika kunci kondisi hadir dalam konteks permintaan, proses kunci seperti yang ditentukan dalam kebijakan. Jika kuncinya

tidak ada, evaluasi elemen ketentuan sebagai benar." Elemen ketentuan lain dalam pernyataan ini masih dapat menghasilkan ketidakcocokkan, tetapi bukan kunci yang hilang saat diperiksa dengan `...IfExists`. Jika Anda menggunakan `"Effect": "Deny"` elemen dengan operator kondisi yang dinegasikan sepertiStringNotEqualsIfExists, permintaan masih ditolak meskipun kunci kondisi tidak ada.

Contoh menggunakan **`IfExists`**

Banyak kunci kondisi menjelaskan informasi tentang tipe sumber daya tertentu dan hanya ada ketika mengakses tipe sumber daya tersebut. Kunci kepatuhan ini tidak ada pada tipe sumber daya lainnya. Hal ini tidak menyebabkan masalah ketika pernyataan kebijakan berlaku hanya pada satu tipe sumber daya. Namun, ada kasus ketika pernyataan tunggal dapat berlaku untuk beberapa tipe sumber daya, seperti ketika pernyataan kebijakan merujuk tindakan dari beberapa layanan atau ketika tindakan tertentu dalam layanan mengakses beberapa tipe sumber daya yang berbeda dalam layanan yang sama. Dalam kasus seperti itu, termasuk kunci kepatuhan yang berlaku hanya pada salah satu sumber daya dalam pernyataan kebijakan dapat menyebabkan elemen `Condition` dalam pernyataan kebijakan gagal sehingga pernyataan `"Effect"` tidak berlaku.

Misalnya, pertimbangkan contoh kebijakan berikut ini:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
      "m3.*"
    ]}}
  }
}
```

Maksud dari kebijakan sebelumnya adalah untuk memungkinkan pengguna meluncurkan instance apa pun yang bertipe`t1`, atau. `t2` `m3` Namun, meluncurkan instance memerlukan akses banyak sumber daya selain instance itu sendiri; misalnya, gambar, pasangan kunci, grup keamanan, dan banyak lagi. Keseluruhan pernyataan dievaluasi terhadap setiap sumber daya yang diperlukan untuk meluncurkan instans. Sumber daya tambahan ini tidak memiliki kunci kondisi `ec2:InstanceType`,

jadi `StringLike` gagal diperiksa, dan pengguna tidak diberi kemampuan untuk meluncurkan tipe instans mana pun.

Untuk mengatasi ini, gunakan operator ketentuan `StringLikeIfExists`. Dengan cara ini, pengujian hanya terjadi jika terdapat kunci kondisi. Anda dapat membaca kebijakan berikut sebagai: “Jika sumber daya yang diperiksa memiliki kunci kondisi`ec2:InstanceType`" ", maka izinkan tindakan hanya jika nilai kunci dimulai dengan`t1.`,`t2.`, atau`m3.`. Jika sumber daya yang diperiksa tidak memiliki kunci kondisi itu, jangan khawatir tentang hal tersebut.” Tanda bintang (*) dalam nilai kunci kondisi, bila digunakan dengan operator `StringLikeIfExists` kondisi, ditafsirkan sebagai wildcard untuk mencapai kecocokan string paral. Pernyataan `DescribeActions` mencakup tindakan yang diperlukan untuk melihat instans di konsol.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RunInstance",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*",
            "Condition": {
                "StringLikeIfExists": {
                    "ec2:InstanceType": [
                        "t1.*",
                        "t2.*",
                        "m3.*"
             ]}}
        },
        {
            "Sid": "DescribeActions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeVpcs",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups"
            ],
            "Resource": "*"
        }]
}
```

Operator ketentuan memeriksa keberadaan kunci kondisi

Gunakan operator `Null` kondisi untuk memeriksa apakah kunci kondisi tidak ada pada saat otorisasi. Dalam pernyataan kebijakan, gunakan `true` (kuncinya tidak ada — yaitu null) atau `false` (kunci ada dan nilainya tidak null).

Anda tidak dapat menggunakan variabel kebijakan dengan operator ketentuan `Null`.

Misalnya, Anda dapat menggunakan operator ketentuan ini untuk menentukan apakah pengguna menggunakan kredensial mereka sendiri untuk operasi atau kredensial sementara. Jika pengguna menggunakan kredensial sementara, maka kuncinya `aws:TokenIssueTime` ada dan memiliki nilai. Contoh berikut menunjukkan kondisi yang menyatakan bahwa pengguna tidak boleh menggunakan kredensi sementara (kunci tidak boleh ada) bagi pengguna untuk menggunakan Amazon. EC2 API

```
{
  "Version": "2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"true"}}
  }
}
```

Kondisi dengan beberapa kunci konteks atau nilai

Anda dapat menggunakan `Condition` elemen kebijakan untuk menguji beberapa kunci konteks atau beberapa nilai untuk satu kunci konteks dalam permintaan. Ketika Anda membuat permintaan untuk AWS, baik secara pemrograman atau melalui AWS Management Console, permintaan Anda mencakup informasi tentang kepala sekolah, operasi, tag, dan lainnya. Anda dapat menggunakan kunci konteks untuk menguji nilai kunci konteks yang cocok dalam permintaan, dengan kunci konteks yang ditentukan dalam kondisi kebijakan. Untuk mempelajari informasi dan data yang disertakan dalam permintaan, lihat Konteks permintaan.

Topik

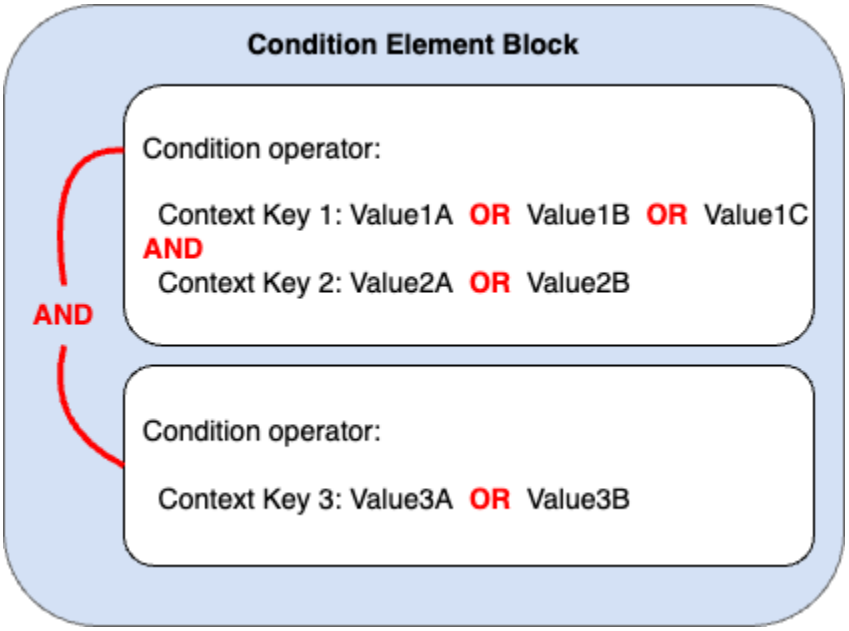
- Logika evaluasi untuk beberapa kunci konteks atau nilai
- Logika evaluasi untuk operator kondisi pencocokan yang dinegasikan

Logika evaluasi untuk beberapa kunci konteks atau nilai

Sebuah `Condition` elemen dapat berisi beberapa operator kondisi, dan setiap operator kondisi dapat berisi beberapa pasangan kunci-nilai konteks. Sebagian besar kunci konteks mendukung penggunaan beberapa nilai, kecuali ditentukan lain.

- Jika pernyataan kebijakan Anda memiliki beberapa operator kondisi, operator kondisi dievaluasi menggunakan logika`AND`.
- Jika pernyataan kebijakan Anda memiliki beberapa kunci konteks yang dilampirkan ke satu operator kondisi, kunci konteks dievaluasi menggunakan logika`AND`.
- Jika operator kondisi tunggal menyertakan beberapa nilai untuk kunci konteks, nilai-nilai tersebut dievaluasi menggunakan logika`OR`.
- Jika operator kondisi pencocokan dinegasikan tunggal menyertakan beberapa nilai untuk kunci konteks, nilai-nilai tersebut dievaluasi menggunakan logika. `NOR`

Semua kunci konteks dalam blok elemen kondisi harus diselesaikan ke true untuk memanggil yang diinginkan `Allow` atau `Deny` efek. Gambar berikut mengilustrasikan logika evaluasi untuk suatu kondisi dengan beberapa operator kondisi dan pasangan kunci-nilai konteks.



Misalnya, kebijakan bucket S3 berikut menggambarkan bagaimana gambar sebelumnya direpresentasikan dalam kebijakan. Blok kondisi mencakup operator kondisi `StringEquals` dan`ArnLike`, dan kunci konteks `aws:PrincipalTag` dan`aws:PrincipalArn`. Untuk memanggil yang diinginkan `Allow` atau `Deny` efek, semua kunci konteks di blok kondisi harus diselesaikan ke

true. Pengguna yang membuat permintaan harus memiliki kunci tag utama, departemen, dan peran, yang menyertakan salah satu nilai kunci tag yang ditentukan dalam kebijakan. Selain itu, prinsipal ARN pengguna yang membuat permintaan harus cocok dengan salah satu `aws:PrincipalArn` nilai yang ditentukan dalam kebijakan untuk dievaluasi sebagai benar.

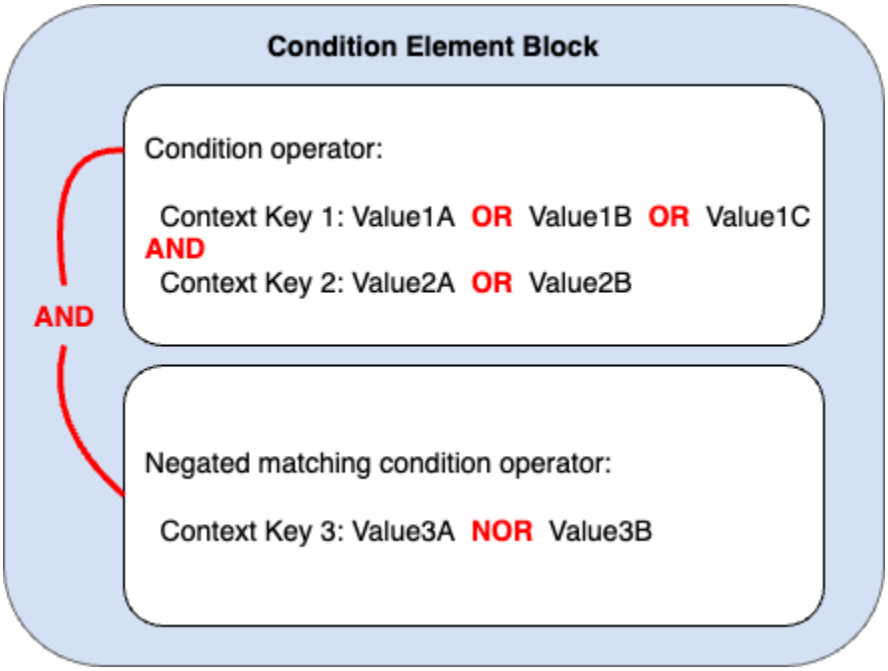
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Logika evaluasi untuk operator kondisi pencocokan yang dinegasikan

Beberapa operator kondisi, seperti `StringNotEquals` atau`ArnNotLike`, menggunakan pencocokan yang dinegasikan untuk membandingkan pasangan nilai kunci konteks dalam kebijakan

Anda dengan pasangan nilai kunci konteks dalam permintaan. Ketika beberapa nilai ditentukan untuk satu kunci konteks dalam kebijakan dengan operator kondisi pencocokan yang dinegasikan, izin efektif berfungsi seperti logika. NOR Dalam pencocokan yang dinegasikan, logis NOR atau NOT OR mengembalikan nilai true hanya jika semua nilai dievaluasi menjadi false.

Gambar berikut mengilustrasikan logika evaluasi untuk suatu kondisi dengan beberapa operator kondisi dan pasangan kunci-nilai konteks. Angka tersebut mencakup operator kondisi pencocokan yang dinegasikan untuk kunci konteks 3.



Misalnya, kebijakan bucket S3 berikut menggambarkan bagaimana gambar sebelumnya direpresentasikan dalam kebijakan. Blok kondisi mencakup operator kondisi `StringEquals` dan`ArnNotLike`, dan kunci konteks `aws:PrincipalTag` dan`aws:PrincipalArn`. Untuk memanggil yang diinginkan `Allow` atau `Deny` efek, semua kunci konteks di blok kondisi harus diselesaikan ke true. Pengguna yang membuat permintaan harus memiliki kunci tag utama, departemen, dan peran, yang menyertakan salah satu nilai kunci tag yang ditentukan dalam kebijakan. Karena operator `ArnNotLike` kondisi menggunakan pencocokan yang dinegasikan, ARN prinsip pengguna yang membuat permintaan tidak boleh cocok dengan `aws:PrincipalArn` nilai yang ditentukan dalam kebijakan yang akan dievaluasi sebagai true.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
```

```
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Kunci konteks bernilai tunggal vs. multivaluasi

Perbedaan antara kunci konteks bernilai tunggal dan multivalued terletak pada jumlah nilai dalam konteks permintaan, bukan jumlah nilai dalam kondisi kebijakan.

- Kunci konteks kondisi bernilai tunggal memiliki paling banyak satu nilai dalam konteks permintaan. Misalnya, saat Anda menandai sumber daya AWS, setiap tag sumber daya disimpan sebagai pasangan nilai kunci. Karena kunci tag sumber daya hanya dapat memiliki satu nilai tag, the section called “ResourceTag” adalah kunci konteks bernilai tunggal. Jangan gunakan operator set kondisi dengan kunci konteks bernilai tunggal.
- Kunci konteks kondisi multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Misalnya, saat Anda menandai sumber daya AWS, Anda dapat menyertakan beberapa pasangan

nilai kunci tag dalam satu permintaan. Oleh karena itu, the section called “TagKeys” adalah kunci konteks multivalued. Kunci konteks multivaluasi memerlukan operator set kondisi.

> **Important**
>
> Kunci konteks multivaluasi memerlukan operator set kondisi. Jangan gunakan operator set kondisi `ForAllValues` atau `ForAnyValue` dengan kunci konteks bernilai tunggal. Untuk mempelajari selengkapnya tentang operator set kondisi, lihatKunci konteks multivaluasi.

Klasifikasi Single-valued dan Multivalued disertakan dalam deskripsi setiap kunci konteks kondisi sebagai tipe Nilai dalam topik. AWS kunci konteks kondisi global Referensi Otorisasi Layanan menggunakan klasifikasi tipe nilai yang berbeda untuk kunci konteks multivalued, menggunakan `ArrayOf` awalan yang diikuti oleh tipe kategori operator kondisi, seperti atau. `ArrayOfString` `ArrayOfARN`

Misalnya, permintaan dapat berasal dari paling banyak satu VPC titik akhir, begitu juga kunci the section called “SourceVpce” konteks bernilai tunggal. Karena layanan dapat memiliki lebih dari satu nama utama layanan yang dimiliki oleh layanan, aws: PrincipalServiceNamesList adalah kunci konteks multivalued.

Anda dapat menggunakan kunci konteks bernilai tunggal yang tersedia sebagai variabel kebijakan, tetapi Anda tidak dapat menggunakan kunci konteks multivaluasi sebagai variabel kebijakan. Untuk informasi lebih lanjut tentang variabel-variable kebijakan, lihat Elemen kebijakan IAM: Variabel dan tanda.

Saat menggunakan kunci konteks yang menyertakan pasangan kunci-nilai, penting untuk dicatat bahwa meskipun mungkin ada beberapa nilai tag-key, masing-masing hanya *`tag-key`* dapat memiliki satu nilai. Oleh karena itu, `aws:RequestTag` dan `aws:ResourceTag` keduanya merupakan kunci konteks bernilai tunggal. Menggunakan operator set kondisi dengan kunci konteks bernilai tunggal dapat menyebabkan kebijakan yang terlalu permisif.

Kunci konteks multivaluasi

Untuk membandingkan kunci konteks kondisi Anda dengan kunci konteks permintaan dengan beberapa nilai, Anda harus menggunakan `ForAllValues` atau `ForAnyValue` mengatur operator. Operator set ini digunakan untuk membandingkan dua set nilai, seperti kumpulan tag dalam permintaan dan kumpulan tag dalam kondisi kebijakan.

Qualifier `ForAllValues` dan `ForAnyValue` qualifier menambahkan fungsionalitas set-operation ke operator kondisi, sehingga Anda dapat menguji kunci konteks permintaan dengan beberapa nilai terhadap beberapa nilai kunci konteks dalam kondisi kebijakan. Selain itu, jika Anda menyertakan kunci konteks string multivalued dalam kebijakan Anda dengan wildcard atau variabel, Anda juga harus menggunakan operator kondisi. `StringLike` Beberapa nilai kunci kondisi harus diapit dalam tanda kurung seperti array, misalnya,. `"Key2":["Value2A", "Value2B"]`

- `ForAllValues`— Qualifier ini menguji apakah nilai setiap anggota dari set permintaan adalah subset dari set kunci konteks kondisi. Kondisi akan ditampilkan `true` jika setiap nilai kunci konteks dalam permintaan cocok dengan setidaknya satu nilai kunci konteks dalam kebijakan. Ini juga mengembalikan `true` jika tidak ada kunci konteks dalam permintaan atau jika nilai kunci konteks menyelesaikan dataset null, seperti string kosong. Untuk mencegah kunci konteks atau kunci konteks yang hilang dengan nilai kosong dievaluasi`true`, Anda dapat menyertakan operator `Null`kondisi dalam kebijakan Anda dengan `false` nilai untuk memeriksa apakah kunci konteks ada dan nilainya bukan null.

  > **Important**
  >
  > Berhati-hatilah jika Anda menggunakan `ForAllValues` dengan `Allow` efek, karena bisa terlalu permisif jika keberadaan kunci konteks yang hilang atau kunci konteks dengan nilai kosong dalam konteks permintaan tidak terduga. Anda dapat menyertakan operator `Null` kondisi dalam kebijakan Anda dengan `false` nilai untuk memeriksa apakah kunci konteks ada dan nilainya bukan null. Sebagai contoh, lihat Mengontrol akses berdasarkan kunci tanda.

- `ForAnyValue`— Kualifikasi ini menguji apakah setidaknya satu anggota dari kumpulan nilai kunci konteks permintaan cocok dengan setidaknya satu anggota kumpulan nilai kunci konteks dalam kondisi kebijakan Anda. Kunci konteks akan kembali `true` jika salah satu nilai kunci konteks dalam permintaan cocok dengan salah satu nilai kunci konteks dalam kebijakan. Untuk tidak ada kunci konteks yang cocok atau dataset nol, kondisi kembali. `false`

> **Note**
>
> Perbedaan antara kunci konteks bernilai tunggal dan multivalued tergantung pada jumlah nilai dalam konteks permintaan, bukan jumlah nilai dalam kondisi kebijakan.

Contoh kebijakan kondisi

Dalam kebijakan IAM, Anda dapat menentukan beberapa nilai untuk kunci konteks bernilai tunggal dan multivalued untuk perbandingan terhadap konteks permintaan. Kumpulan contoh kebijakan berikut menunjukkan kondisi kebijakan dengan beberapa kunci konteks dan nilai.

> **Note**
>
> Jika Anda ingin memberikan sebuah kebijakan untuk disertakan dalam panduan referensi ini, gunakan tombol Umpan Balik di bagian bawah halaman ini. Untuk contoh kebijakan berbasis identitas IAM, lihat. Contoh kebijakan berbasis identitas IAM

Contoh kebijakan kondisi: Kunci konteks bernilai tunggal

- Beberapa blok kondisi dengan kunci konteks bernilai tunggal. (Lihat contoh ini.)
- Satu blok kondisi dengan beberapa kunci dan nilai konteks bernilai tunggal. (Lihat contoh ini.)

Contoh kebijakan kondisi: Kunci konteks berbilang nilai

- Tolak kebijakan dengan operator set kondisi`ForAllValues`. (Lihat contoh ini.)
- Tolak kebijakan dengan operator set kondisi`ForAnyValue`. (Lihat contoh ini.)

Contoh kunci konteks multivaluasi

Kumpulan contoh kebijakan berikut menunjukkan cara membuat kondisi kebijakan dengan kunci konteks multivalued.

Contoh: Tolak kebijakan dengan operator set kondisi ForAllValues

Contoh kebijakan berbasis identitas berikut menyangkal penggunaan tindakan penandaan IAM saat awalan kunci tag tertentu disertakan dalam permintaan. Setiap nilai untuk kunci konteks `aws:TagKeys` menyertakan wildcard (*) untuk pencocokan string parsi. Kebijakan ini menyertakan operator yang `ForAllValues` disetel dengan kunci konteks `aws:TagKeys` karena kunci konteks permintaan dapat menyertakan beberapa nilai. Agar kunci konteks `aws:TagKeys` mengembalikan nilai true, setiap nilai dalam permintaan harus cocok dengan setidaknya satu nilai dalam kebijakan.

Operator `ForAllValues` set juga mengembalikan true jika tidak ada kunci konteks dalam permintaan, atau jika nilai kunci konteks menyelesaikan ke dataset null, seperti string kosong. Untuk

mencegah kunci konteks atau kunci konteks yang hilang dengan nilai kosong dievaluasi menjadi true, sertakan operator `Null` kondisi dalam kebijakan Anda dengan nilai `false` untuk memeriksa apakah kunci konteks dalam permintaan ada dan nilainya bukan null.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Null": {
          "aws:TagKeys": "false"
        },
        "ForAllValues:StringLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*",
            "key3*"
          ]
        }
      }
    }
  ]
}
```

Contoh: Tolak kebijakan dengan operator set kondisi ForAnyValue

Contoh kebijakan berbasis identitas berikut menyangkal pembuatan snapshot volume instans EC2 jika ada snapshot yang ditandai dengan salah satu kunci tag yang ditentukan dalam kebijakan, atau. `environment` `webserver` Kebijakan ini menyertakan operator yang `ForAnyValue` disetel dengan kunci konteks `aws:TagKeys` karena kunci konteks permintaan dapat menyertakan beberapa nilai. Jika permintaan penandaan Anda menyertakan salah satu nilai kunci tag yang ditentukan dalam kebijakan, kunci `aws:TagKeys` konteks akan mengembalikan nilai true dengan mengaktifkan efek kebijakan penolakan.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": ["environment", "webserver"]
                }
            }
        }
    ]
}
```

Contoh kebijakan kunci konteks bernilai tunggal

Kumpulan contoh kebijakan berikut menunjukkan cara membuat kondisi kebijakan dengan kunci konteks bernilai tunggal.

Contoh: Beberapa blok kondisi dengan kunci konteks bernilai tunggal

Ketika blok kondisi memiliki beberapa kondisi, masing-masing dengan satu kunci konteks, semua kunci konteks harus diselesaikan ke true agar `Deny` efek yang diinginkan `Allow` atau dipanggil. Saat Anda menggunakan operator kondisi pencocokan yang dinegasikan, logika evaluasi nilai kondisi dibalik.

Contoh berikut memungkinkan pengguna membuat EC2 volume dan menerapkan tag ke volume selama pembuatan volume. Konteks permintaan harus menyertakan nilai untuk kunci konteks`aws:RequestTag/project`, dan nilai untuk kunci konteks `aws:ResourceTag/environment` dapat berupa apa saja kecuali produksi.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:::volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:region:account:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

Konteks permintaan harus menyertakan nilai tag proyek dan tidak dapat dibuat untuk sumber daya produksi untuk memanggil efek. `Allow` EC2Volume berikut berhasil dibuat karena nama proyek `Feature3` dengan tag `QA` sumber daya.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

Contoh: Satu blok kondisi dengan beberapa kunci dan nilai konteks bernilai tunggal

Ketika blok kondisi berisi beberapa kunci konteks dan setiap kunci konteks memiliki beberapa nilai, setiap kunci konteks harus menyelesaikan ke true untuk setidaknya satu nilai kunci untuk yang diinginkan `Allow` atau `Deny` efek yang akan dipanggil. Saat Anda menggunakan operator kondisi pencocokan yang dinegasikan, logika evaluasi nilai kunci konteks dibalik.

Contoh berikut memungkinkan pengguna untuk memulai dan menjalankan tugas di Amazon Elastic Container Service cluster.

- Konteks permintaan harus menyertakan `production` OR `prod-backup` untuk kunci `aws:RequestTag/environment` konteks AND.
- Kunci `ecs:cluster` konteks memastikan bahwa tugas dijalankan di salah satu `default2` ARN ECS kluster `default1` OR.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
```

```
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
        }
      }
    }
  ]
}
```

Elemen kebijakan IAM: Variabel dan tanda

Gunakan AWS Identity and Access Management (IAM) variabel kebijakan sebagai placeholder ketika Anda tidak mengetahui nilai pasti dari sumber daya atau kunci kondisi saat Anda menulis kebijakan.

> **Note**
>
> Jika AWS tidak dapat menyelesaikan variabel ini dapat menyebabkan seluruh pernyataan menjadi tidak valid. Misalnya, jika Anda menggunakan `aws:TokenIssueTime` variabel, variabel menyelesaikan ke nilai hanya ketika pemohon diautentikasi menggunakan kredensi sementara (peran). IAM Untuk mencegah variabel menyebabkan pernyataan tidak valid, gunakan... IfExists operator kondisi.

Topik

- Pengantar
- Menggunakan variabel dalam kebijakan
- Tanda sebagai variabel kebijakan
- Tempat Anda dapat menggunakan variabel kebijakan
- Variabel kebijakan tanpa nilai
- Meminta informasi yang dapat Anda gunakan untuk variabel kebijakan
- Menentukan nilai default

- Untuk informasi selengkapnya

Pengantar

Dalam IAM kebijakan, banyak tindakan memungkinkan Anda memberikan nama untuk sumber daya tertentu yang ingin Anda kendalikan aksesnya. Misalnya, kebijakan berikut memungkinkan pengguna untuk membuat daftar, membaca, dan menulis objek di bucket S3 `amzn-s3-demo-bucket` untuk `marketing` proyek.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/marketing/*"]
    }
  ]
}
```

Dalam beberapa kasus, Anda mungkin tidak mengetahui nama pasti sumber daya saat Anda menyusun kebijakan. Anda mungkin ingin melakukan generalisasi kebijakan sehingga dapat digunakan oleh banyak pengguna tanpa harus membuat salinan unik dari kebijakan tersebut untuk setiap pengguna. Alih-alih membuat kebijakan terpisah untuk setiap pengguna, kami sarankan Anda membuat kebijakan grup tunggal yang berfungsi untuk setiap pengguna dalam grup tersebut.

Menggunakan variabel dalam kebijakan

Anda dapat menentukan nilai dinamis di dalam kebijakan dengan menggunakan variabel kebijakan yang menetapkan placeholder dalam kebijakan.

Variabel ditandai menggunakan **$**awalan diikuti oleh sepasang kurawal kurawal (**{ }**) yang menyertakan nama variabel nilai dari permintaan.

Ketika kebijakan dievaluasi, variabel kebijakan diganti dengan nilai yang berasal dari kunci konteks bersyarat yang diteruskan dalam permintaan. Variabel dapat digunakan dalam kebijakan berbasis identitas, kebijakan sumber daya, kebijakan kontrol layanan, kebijakan sesi, dan VPC kebijakan titik akhir. Kebijakan berbasis identitas yang digunakan sebagai batas izin juga mendukung variabel kebijakan.

Kunci konteks kondisi global dapat digunakan sebagai variabel dalam permintaan di seluruh AWS layanan. Kunci kondisi khusus layanan juga dapat digunakan sebagai variabel saat berinteraksi dengan AWS sumber daya, tetapi hanya tersedia ketika permintaan dibuat terhadap sumber daya yang mendukungnya. Untuk daftar kunci konteks yang tersedia untuk masing-masing AWS layanan dan sumber daya, lihat Referensi Otorisasi Layanan. Dalam keadaan tertentu, Anda tidak dapat mengisi kunci konteks kondisi global dengan nilai. Untuk mempelajari lebih lanjut tentang setiap kunci, lihat AWS kunci konteks kondisi global.

> **Important**
>
> - Nama utama peka dengan huruf besar dan kecil. Misalnya, `aws:CurrentTime` setara dengan `AWS:currenttime`.
> - Anda dapat menggunakan kunci kondisi bernilai tunggal apa pun sebagai variabel. Anda tidak dapat menggunakan kunci kondisi multivalued sebagai variabel.

Contoh berikut menunjukkan kebijakan untuk IAM peran atau pengguna yang menggantikan nama sumber daya tertentu dengan variabel kebijakan. Anda dapat menggunakan kembali kebijakan ini dengan memanfaatkan kunci `aws:PrincipalTag` kondisi. Jika kebijakan ini dievaluasi, hanya `${aws:PrincipalTag/team}` mengizinkan tindakan jika nama bucket diakhiri dengan nama tim dari tag `team` utama.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/team}/*"]
    }
  ]
}
```

Variabel ditandai menggunakan prefiks $ yang diikuti dengan sepasang rangka yang melengkung ({ }). Di dalam ${ } karakter, Anda dapat menyertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan. Nilai-nilai yang dapat Anda gunakan akan dibahas nanti di halaman ini.

Untuk detail tentang kunci kondisi global ini, lihat aws:PrincipalTag/tag-kunci di daftar kunci kondisi global.

> **Note**
>
> Untuk menggunakan variabel kebijakan, Anda harus menyertakan elemen `Version` dalam pernyataan, dan versinya harus diatur ke versi yang mendukung variabel kebijakan. Variabel diperkenalkan dalam versi `2012-10-17`. Versi sebelumnya dari bahasa kebijakan tidak mendukung variabel kebijakan. Jika Anda tidak menyertakan elemen `Version` dan mengaturnya ke tanggal versi yang sesuai, variabel seperti `${aws:username}` diperlakukan sebagai string literal dalam kebijakan ini.
> Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan, di sisi lain, dibuat saat Anda mengubah kebijakan yang dikelola pelangganIAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola. Untuk mempelajari selengkapnya tentang elemen kebijakan `Version`, lihat the section called “Version”. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called “Kebijakan IAM versioning”.

Kebijakan yang memungkinkan prinsipal untuk mendapatkan objek dari jalur /David dari bucket S3 terlihat seperti ini:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3::amzn-s3-demo-bucket/David/*"]
  }]
}
```

Jika kebijakan ini dilampirkan ke penggunaDavid, pengguna tersebut mendapatkan objek dari bucket S3-nya sendiri, tetapi Anda harus membuat kebijakan terpisah untuk setiap pengguna yang menyertakan nama pengguna. Kemudian, Anda akan menerapkan setiap kebijakan kepada pengguna individu.

Dengan menggunakan variabel kebijakan, Anda dapat membuat kebijakan yang dapat digunakan kembali. Kebijakan berikut memungkinkan pengguna mendapatkan objek dari bucket Amazon S3 jika nilai tag-key aws:PrincipalTag cocok dengan nilai kunci tag yang diteruskan dalam permintaanowner.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/owner": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

Bila Anda menggunakan variabel kebijakan sebagai pengganti pengguna seperti ini, Anda tidak harus memiliki kebijakan terpisah untuk setiap pengguna individu. Dalam contoh berikut, kebijakan dilampirkan pada IAM peran yang diasumsikan oleh Manajer Produk menggunakan kredensyal keamanan sementara. Saat pengguna membuat permintaan untuk menambahkan objek Amazon S3, IAM ganti nilai dept tag dari permintaan ${aws:PrincipalTag} variabel saat ini dan evaluasi kebijakan.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOnlyDeptS3Prefix",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/dept}/*"],
    }
  ]
}
```

Tanda sebagai variabel kebijakan

Di beberapa AWS layanan Anda dapat melampirkan atribut kustom Anda sendiri ke sumber daya yang dibuat oleh layanan tersebut. Misalnya, Anda dapat menerapkan tag ke bucket Amazon S3 atau ke pengguna. IAM Tanda ini adalah pasangan nilai kunci. Anda menentukan nama kunci tanda dan nilai yang terkait dengan nama kunci tersebut. Misalnya, Anda dapat membuat tanda dengan kunci **`department`** dan nilai **`Human Resources`**. Untuk informasi selengkapnya tentang menandai IAM entitas, lihatTag untuk AWS Identity and Access Management sumber daya. Untuk informasi tentang penandaan sumber daya yang dibuat oleh orang lain AWS layanan, lihat dokumentasi untuk layanan itu. Untuk informasi tentang menggunakan Editor Tag, lihat Bekerja dengan Editor Tag di AWS Management Console Panduan Pengguna.

Anda dapat menandai IAM sumber daya untuk menyederhanakan menemukan, mengatur, dan melacak sumber daya AndaIAM. Anda juga dapat menandai IAM identitas untuk mengontrol akses ke sumber daya atau menandai dirinya sendiri. Untuk mempelajari selengkapnya tentang penggunaan tanda untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Tempat Anda dapat menggunakan variabel kebijakan

Anda dapat menggunakan variabel kebijakan di elemen `Resource` dan dalam perbandingan string dalam elemen `Condition`.

Elemen sumber daya

Anda dapat menggunakan variabel kebijakan dalam `Resource` elemen, tetapi hanya di bagian sumber dayaARN. Bagian ini ARN muncul setelah titik lima (:). Anda tidak dapat menggunakan variabel untuk mengganti bagian ARN sebelum titik lima, seperti layanan atau akun. Untuk informasi selengkapnya tentang ARN format, lihatIAM ARNs.

Untuk mengganti bagian dari a ARN dengan nilai tag, kelilingi awalan dan nama kunci dengan. `${ }` Misalnya, elemen Resource berikut hanya mengacu pada bucket yang diberi nama sama dengan nilai dalam tag departemen pengguna yang meminta.

```
"Resource": ["arn:aws::s3:::amzn-s3-demo-bucket/
${aws:PrincipalTag/department}"]
```

Banyak AWS penggunaan sumber daya ARNs yang berisi nama yang dibuat pengguna. IAMKebijakan berikut memastikan bahwa hanya pengguna yang dituju dengan nilai tag access-project, access-application, dan access-environment yang cocok yang dapat memodifikasi resource mereka. Selain itu, menggunakan kecocokan wildcard *, mereka dapat mengizinkan sufiks nama sumber daya khusus.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
      ]
    }
  ]
}
```

Elemen kondisi

Anda dapat menggunakan variabel kebijakan untuk `Condition` nilai dalam kondisi apa pun yang melibatkan operator string atau ARN operator. Operator string termasuk `StringEquals`, `StringLike`, dan `StringNotLike`. ARNoperator termasuk `ArnEquals` dan`ArnLike`. Anda tidak dapat menggunakan variabel kebijakan dengan operator lainnya, seperti operator `Numeric`, `Date`, `Boolean`, `Binary`, `IP Address`, atau `Null`. Untuk informasi selengkapnya tentang operator kondisi, lihat IAMJSONelemen kebijakan: Operator kondisi.

Saat mereferensikan tanda di dalam ekspresi elemen `Condition`, gunakan prefiks dan nama kunci yang relevan sebagai kunci kondisi. Kemudian gunakan nilai yang ingin Anda uji di dalam nilai kondisi.

Misalnya, contoh kebijakan berikut memungkinkan akses penuh ke pengguna, tetapi hanya jika tag `costCenter` dilampirkan ke pengguna. Tanda harus memiliki nilai `12345` atau `67890`. Jika tanda tidak memiliki nilai, atau memiliki nilai lain, permintaan akan gagal.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Variabel kebijakan tanpa nilai

Ketika variabel kebijakan mereferensikan kunci konteks kondisi yang tidak memiliki nilai atau tidak ada dalam konteks otorisasi untuk permintaan, nilainya secara efektif nol. Tidak ada nilai yang sama atau serupa. Kunci konteks kondisi mungkin tidak ada dalam konteks otorisasi ketika:

- Anda menggunakan kunci konteks kondisi khusus layanan dalam permintaan ke sumber daya yang tidak mendukung kunci kondisi tersebut.
- Tag pada IAM prinsipal, sesi, sumber daya, atau permintaan tidak ada.
- Keadaan lain seperti yang tercantum untuk setiap konteks kondisi global masukAWS kunci konteks kondisi global.

Bila Anda menggunakan variabel tanpa nilai dalam elemen kondisi IAM kebijakan, IAMJSONelemen kebijakan: Operator kondisi suka `StringEquals` atau `StringLike` tidak cocok, dan pernyataan kebijakan tidak berlaku.

Operator kondisi terbalik `StringNotLike` menyukai `StringNotEquals` atau mencocokkan dengan nilai nol, karena nilai kunci kondisi yang mereka uji tidak sama dengan atau seperti nilai nol efektif.

Dalam contoh berikut, `aws:principaltag/Team` harus sama dengan `s3:ExistingObjectTag/Team` mengizinkan akses. Akses ditolak secara eksplisit ketika tidak `aws:principaltag/Team` disetel. Jika variabel yang tidak memiliki nilai dalam konteks otorisasi digunakan sebagai bagian dari `Resource` atau `NotResource` elemen kebijakan, sumber daya yang menyertakan variabel kebijakan tanpa nilai tidak akan cocok dengan sumber daya apa pun.

```
{
  "Version": "2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
      }
    }
  ]
}
```

Meminta informasi yang dapat Anda gunakan untuk variabel kebijakan

Anda dapat menggunakan `Condition` elemen JSON kebijakan untuk membandingkan kunci dalam konteks permintaan dengan nilai kunci yang Anda tentukan dalam kebijakan Anda. Saat Anda menggunakan variabel kebijakan, AWS mengganti nilai dari kunci konteks permintaan sebagai pengganti variabel dalam kebijakan Anda.

Nilai-nilai kunci utama

Nilai untuk `aws:username`, `aws:userid`, dan `aws:PrincipalType` bergantung pada tipe prinsipal yang memulai permintaan. Misalnya, permintaan dapat dibuat menggunakan kredensi IAM pengguna, IAM peran, atau Pengguna root akun AWS. Daftar berikut menunjukkan nilai untuk kunci ini untuk berbagai jenis prinsipal.

- Pengguna root akun AWS

  - `aws:username`: (tidak ada)
  - `aws:userid`: Akun AWS ID
  - `aws:PrincipalType`: `Account`
- IAMpengguna
  - `aws:username`: *`IAM-user-name`*
  - `aws:userid`: ID unik
  - `aws:PrincipalType`: `User`
- Pengguna federasi
  - `aws:username`: (tidak ada)
  - `aws:userid`: *`account`*:*`caller-specified-name`*
  - `aws:PrincipalType`: `FederatedUser`
- Pengguna federasi web dan pengguna SAML federasi

> **Note**
>
> Untuk informasi tentang kunci kebijakan yang tersedia saat Anda menggunakan OIDC federasi, lihatOIDCfederasi.

  - `aws:username`: (tidak ada)
  - `aws:userid`: (tidak ada)
  - `aws:PrincipalType`: `AssumedRole`
- Peran yang diasumsikan
  - `aws:username`: (tidak ada)
  - `aws:userid`: *`role-id`*:*`caller-specified-role-name`*
  - `aws:PrincipalType`: `Assumed role`
- Peran yang ditetapkan ke EC2 instans Amazon
  - `aws:username`: (tidak ada)
  - `aws:userid`: *`role-id`*:*`ec2-instance-id`*
  - `aws:PrincipalType`: `Assumed role`
- Penelepon anonim (hanya Amazon SQS Amazon SNS dan Amazon S3)
  - `aws:username`: (tidak ada)

- `aws:userid`: (tidak ada)
- `aws:PrincipalType`: `Anonymous`

Untuk item dalam daftar ini, ingat hal berikut:

- tidak ada berarti bahwa nilai tersebut tidak tercantum dalam informasi permintaan saat ini, dan setiap upaya untuk mencocokannya gagal dan menyebabkan pernyataan tersebut menjadi tidak valid.
- *`role-id`* adalah pengidentifikasi unik yang ditetapkan untuk setiap peran saat dibuat. Anda dapat menampilkan ID peran dengan AWS CLI perintah: `aws iam get-role --role-name` *`rolename`*
- *`caller-specified-name`* and *`caller-specified-role-name`* adalah nama yang diberikan oleh proses panggilan (seperti aplikasi atau layanan) ketika membuat panggilan untuk mendapatkan kredensial sementara.
- *`ec2-instance-id`* adalah nilai yang ditetapkan ke instance saat diluncurkan dan muncul di halaman Instans EC2 konsol Amazon. Anda juga dapat menampilkan ID instance dengan menjalankan AWS CLI perintah: `aws ec2 describe-instances`

Informasi yang tersedia dalam permintaan bagi pengguna gabungan

Pengguna federasi adalah pengguna yang diautentikasi menggunakan sistem selain. IAM Misalnya, perusahaan mungkin memiliki aplikasi untuk digunakan di rumah yang melakukan panggilan ke AWS. Mungkin tidak praktis untuk memberikan IAM identitas kepada setiap pengguna perusahaan yang menggunakan aplikasi. Sebagai gantinya, perusahaan mungkin menggunakan aplikasi proxy (tingkat menengah) yang memiliki IAM identitas tunggal, atau perusahaan mungkin menggunakan penyedia SAML identitas (iDP). Aplikasi proxy atau SAML IDP mengautentikasi pengguna individu menggunakan jaringan perusahaan. Aplikasi proxy kemudian dapat menggunakan IAM identitasnya untuk mendapatkan kredensyal keamanan sementara untuk pengguna individu. SAMLIDP pada dasarnya dapat bertukar informasi identitas untuk AWS kredensil keamanan sementara. Kredensi sementara kemudian dapat digunakan untuk mengakses AWS sumber daya.

Demikian pula, Anda dapat membuat aplikasi untuk perangkat seluler yang perlu diakses aplikasi AWS sumber daya. Dalam hal ini, Anda dapat menggunakan OIDCfederasi, di mana aplikasi mengautentikasi pengguna menggunakan penyedia identitas terkenal seperti Login with Amazon, Amazon Cognito, Facebook, atau Google. Aplikasi kemudian dapat menggunakan informasi

otentikasi pengguna dari penyedia ini untuk mendapatkan kredensyal keamanan sementara untuk mengakses AWS sumber daya.

Cara yang disarankan untuk menggunakan OIDC federasi adalah dengan memanfaatkan Amazon Cognito dan AWS ponselSDKs. Untuk informasi selengkapnya, lihat berikut ini:

- Panduan Pengguna Amazon Cognito
- Skenario umum untuk kredensial sementara

Karakter khusus

Ada beberapa variabel kebijakan khusus yang telah ditetapkan sebelumnya yang memiliki nilai tetap yang memungkinkan Anda mewakili karakter yang memiliki arti khusus. Jika karakter khusus ini adalah bagian dari string, Anda mencoba untuk mencocokkannya dan Anda memasukkannya secara literal sehingga mereka akan disalahartikan. Misalnya, menyisipkan tanda bintang * dalam string akan ditafsirkan sebagai wildcard yang cocok dengan karakter apa pun, bukan sebagai tanda bintang * yang sebenarnya. Dalam kasus ini, Anda dapat menggunakan variabel kebijakan yang telah ditentukan sebelumnya berikut ini:

- ${*} - gunakan jika Anda memerlukan karakter * (tanda bintang).
- ${?} - gunakan jika Anda memerlukan katakter ? (tanda tanya).
- ${$} - gunakan jika Anda memerlukan karakter $ (simbol dolar).

Variabel kebijakan yang sudah ditentukan sebelumnya ini dapat digunakan dalam string apa pun tempat Anda dapat menggunakan variabel kebijakan reguler.

Menentukan nilai default

Untuk menambahkan nilai default ke variabel, sertai nilai default dengan tanda kutip tunggal (`' '`), dan pisahkan teks variabel serta nilai default dengan koma dan spasi (`, `).

Misalnya, jika prinsipal diberi tag`team=yellow`, mereka dapat mengakses bucket `ExampleCorp's` Amazon S3 bernama. `amzn-s3-demo-bucket-yellow` Kebijakan dengan sumber daya ini memungkinkan anggota tim untuk mengakses keranjang tim mereka, tetapi tidak dengan tim lain. Untuk pengguna tanpa tag tim, ini menetapkan nilai default `company-wide` untuk nama bucket. Pengguna ini hanya dapat mengakses `amzn-s3-demo-bucket-company-wide` bucket di mana mereka dapat melihat informasi yang luas, seperti instruksi untuk bergabung dengan tim.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Untuk informasi selengkapnya

Untuk informasi selengkapnya tentang kebijakan, lihat hal berikut:

- Kebijakan dan izin di AWS Identity and Access Management
- Contoh kebijakan berbasis identitas IAM
- IAMJSONreferensi elemen kebijakan
- Logika evaluasi kebijakan
- OIDCfederasi

Elemen kebijakan IAM JSON: Tipe data yang didukung

Bagian ini mencantumkan tipe data yang didukung ketika Anda menentukan nilai dalam kebijakan JSON. Bahasa kebijakan tidak mendukung semua tipe untuk setiap elemen kebijakan; untuk informasi tentang setiap elemen, lihat bagian sebelumnya.

- String
- Bilangan (Bulat dan Desimal)
- Boolean
- Nol
- Daftar
- Peta
- Struktur (yang hanya berupa Peta bersarang)

Tabel berikut memetakan setiap tipe data ke serialisasi. Perhatikan bahwa semua kebijakan harus dalam UTF-8. Untuk informasi tentang tipe data JSON, kunjungi RFC 4627.

Tipe	JSON
String	String
Bulat	Bilangan

Tipe	JSON
Desimal	Bilangan
Boolean	salah benar
Nol	nol
Tanggal	String sesuai dengan Profil W3C ISO 8601
IpAddress	String sesuai dengan RFC 4632
Daftar	Susunan
Objek	Objek

Logika evaluasi kebijakan

Ketika seorang kepala sekolah mencoba menggunakan AWS Management Console, AWS API, atau AWS CLI, kepala sekolah itu mengirimkan permintaan ke AWS. Ketika sebuah AWS layanan menerima permintaan, AWS menyelesaikan beberapa langkah untuk menentukan apakah akan mengizinkan atau menolak permintaan.

1. Otentikasi - AWS pertama mengotentikasi kepala sekolah yang membuat permintaan, jika perlu. Langkah ini tidak diperlukan untuk beberapa layanan, seperti Amazon S3 yang memungkinkan beberapa permintaan dari pengguna anonim.
2. Memproses konteks permintaan – AWS memproses informasi yang dikumpulkan dalam permintaan untuk menentukan kebijakan mana yang berlaku untuk permintaan tersebut.
3. Mengevaluasi kebijakan dalam satu akun – AWS mengevaluasi semua jenis kebijakan, yang mempengaruhi urutan di mana kebijakan dievaluasi.
4. Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun. – AWS kemudian memproses kebijakan terhadap konteks permintaan untuk menentukan apakah permintaan diizinkan atau ditolak.

Memproses konteks permintaan

AWS memproses permintaan untuk mengumpulkan informasi berikut ke dalam konteks permintaan:

- Tindakan (atau operasi) – Tindakan atau operasi yang ingin dilakukan oleh prinsipal.
- Sumber Daya — The AWS objek sumber daya di mana tindakan atau operasi dilakukan.
- Prinsipal – Pengguna, peran, pengguna gabungan, atau aplikasi yang mengirimkan permintaan tersebut. Informasi tentang prinsipal mencakup kebijakan yang terkait dengan prinsipal tersebut
- Data lingkungan — Informasi tentang alamat IP, agen pengguna, status yang SSL diaktifkan, atau waktu hari.
- Data sumber daya – Data terkait sumber daya yang diminta. Ini dapat mencakup informasi seperti nama tabel DynamoDB atau tag pada instance Amazon. EC2

AWS kemudian menggunakan informasi ini untuk menemukan kebijakan yang berlaku untuk konteks permintaan.

Mengevaluasi kebijakan dalam satu akun

Bagaimana AWS mengevaluasi kebijakan tergantung pada jenis kebijakan yang berlaku untuk konteks permintaan. Jenis kebijakan berikut, yang tercantum dalam urutan frekuensi, tersedia untuk digunakan dalam satu Akun AWS. Untuk informasi selengkapnya tentang jenis kebijakan ini, lihatKebijakan dan izin di AWS Identity and Access Management. Untuk mempelajari caranya AWS mengevaluasi kebijakan untuk akses lintas akun, lihat. Logika evaluasi kebijakan lintas akun

1. Kebijakan berbasis identitas — Kebijakan berbasis identitas dilampirkan pada IAM identitas (pengguna, grup pengguna, atau peran) dan memberikan izin kepada IAM entitas (pengguna dan peran). Jika hanya kebijakan berbasis identitas yang berlaku untuk permintaan, maka AWS memeriksa semua kebijakan tersebut untuk setidaknya satu`Allow`.
2. Kebijakan berbasis sumber daya — Kebijakan berbasis sumber daya memberikan izin kepada prinsipal (akun, pengguna, peran, dan kepala sesi seperti sesi peran dan pengguna gabungan) yang ditentukan sebagai prinsipal. IAM Izin menentukan apa yang dapat dilakukan oleh prinsipal dengan sumber daya yang memiliki kebijakan tersebut. Jika kebijakan berbasis sumber daya dan kebijakan berbasis identitas berlaku untuk permintaan, maka AWS memeriksa semua kebijakan untuk setidaknya satu`Allow`. Ketika kebijakan berbasis sumber daya dievaluasi, prinsipal ARN yang ditentukan dalam kebijakan menentukan apakah penolakan implisit dalam jenis kebijakan lain berlaku untuk keputusan akhir.
3. IAMbatas izin — Batas izin adalah fitur lanjutan yang menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada IAM entitas (pengguna atau peran). Saat Anda menetapkan batas izin untuk suatu entitas, entitas hanya dapat melakukan tindakan yang diizinkan oleh kedua kebijakan berbasis identitas dan batas izinnya. Dalam beberapa kasus, penolakan

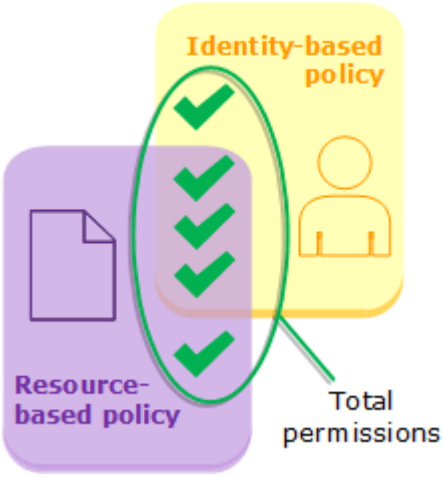
implisit dalam batas izin dapat membatasi izin yang diberikan oleh kebijakan berbasis sumber daya. Untuk mempelajari lebih lanjut, lihat Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun. nanti dalam topik ini.

4. AWS Organizations kebijakan kontrol layanan (SCPs) — Organizations SCPs menentukan izin maksimum untuk organisasi atau unit organisasi (OU). SCPMaksimum berlaku untuk prinsipal di akun anggota, termasuk masing-masing Pengguna root akun AWS. Jika SCP ada, kebijakan berbasis identitas dan sumber daya memberikan izin kepada kepala sekolah di akun anggota hanya jika kebijakan tersebut dan mengizinkan tindakan tersebut. SCP Jika ada batas izin dan batas izin, maka batas, kebijakan berbasis identitas SCP harus mengizinkan tindakan tersebut. SCP
5. Kebijakan sesi – Kebijakan sesi adalah kebijakan tingkat lanjut yang Anda teruskan sebagai parameter saat Anda secara terprogram membuat sesi sementara untuk pengguna peran atau pengguna gabungan. Untuk membuat sesi peran secara terprogram, gunakan salah satu operasi. `AssumeRole*` API Saat Anda melakukan ini dan meneruskan kebijakan sesi, izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas IAM entitas dan kebijakan sesi. Untuk membuat sesi pengguna federasi, Anda menggunakan kunci akses IAM pengguna untuk memanggil operasi secara terprogram. `GetFederationToken` API Kebijakan berbasis sumber daya memiliki efek yang berbeda pada evaluasi izin kebijakan sesi. Perbedaannya tergantung pada apakah pengguna atau peran ARN atau sesi terdaftar sebagai prinsipal dalam kebijakan berbasis sumber daya. ARN Untuk informasi selengkapnya, lihat Kebijakan sesi.

Ingat, penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.

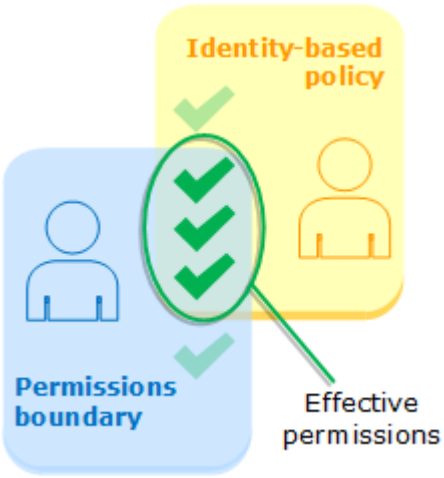
Mengevaluasi kebijakan berbasis identitas dengan kebijakan berbasis sumber daya

Kebijakan berbasis identitas dan kebijakan berbasis sumber daya memberikan izin kepada identitas atau sumber daya yang melekat padanya. Ketika IAM entitas (pengguna atau peran) meminta akses ke sumber daya dalam akun yang sama, AWS mengevaluasi semua izin yang diberikan oleh kebijakan berbasis identitas dan sumber daya. Izin yang dihasilkan adalah izin total dari dua tipe. Jika suatu tindakan diizinkan oleh kebijakan berbasis identitas, kebijakan berbasis sumber daya, atau keduanya, maka AWS memungkinkan tindakan. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.



Mengevaluasi kebijakan berbasis identitas dengan batas izin

Saat AWS mengevaluasi kebijakan berbasis identitas dan batas izin untuk pengguna, izin yang dihasilkan adalah persimpangan dari dua kategori. Artinya ketika Anda menambahkan batas izin ke pengguna dengan kebijakan berbasis identitas yang ada, Anda mungkin mengurangi tindakan yang dapat dilakukan pengguna. Atau, saat Anda menghapus batas izin dari pengguna, Anda mungkin meningkatkan tindakan yang dapat mereka lakukan. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin. Untuk melihat informasi tentang cara tipe kebijakan lain dievaluasi dengan batas izin, lihat Mengevaluasi izin efektif dengan batasan.



Mengevaluasi kebijakan berbasis identitas dengan Organizations SCPs

Ketika pengguna termasuk dalam akun yang merupakan anggota organisasi, izin yang dihasilkan adalah persimpangan dari kebijakan pengguna dan. SCP Ini berarti bahwa suatu tindakan harus diizinkan oleh kebijakan berbasis identitas dan. SCP Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.

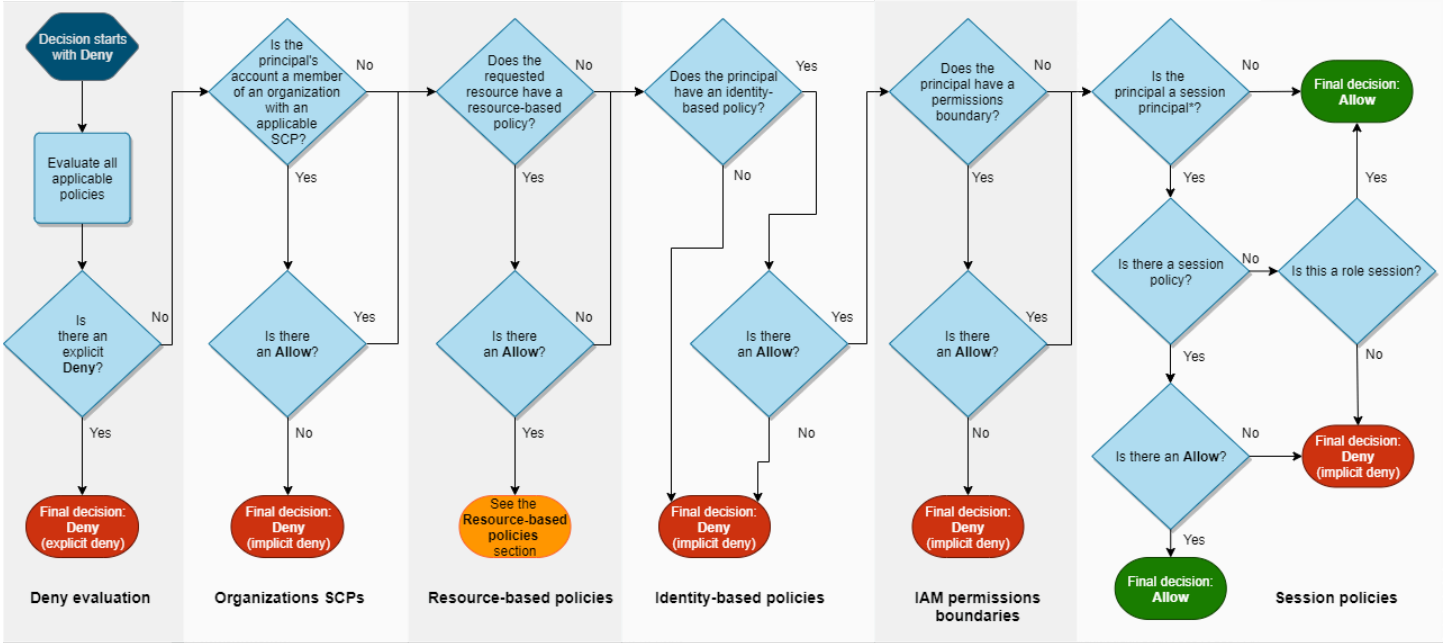
Anda dapat mengetahui apakah akun Anda adalah anggota organisasi di AWS Organizations. Anggota organisasi mungkin terpengaruh olehSCP. Untuk melihat data ini menggunakan AWS CLI perintah atau AWS APIoperasi, Anda harus memiliki izin untuk `organizations:DescribeOrganization` tindakan untuk entitas Organizations Anda. Anda harus memiliki izin tambahan untuk melakukan operasi di konsol Organisasi. Untuk mengetahui SCP apakah menolak akses ke permintaan tertentu, atau untuk mengubah izin efektif Anda, hubungi AWS Organizations administrator.

Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun.

Asumsikan bahwa kepala sekolah mengirimkan permintaan ke AWS untuk mengakses sumber daya dalam akun yang sama dengan entitas prinsipal. Bagian AWS Kode penegakan memutuskan apakah permintaan harus diizinkan atau ditolak. AWS mengevaluasi semua kebijakan yang berlaku untuk konteks permintaan. Berikut ini adalah ringkasan dari AWS logika evaluasi untuk kebijakan dalam satu akun.

- Secara default, semua permintaan ditolak secara implisit dengan pengecualian Pengguna root akun AWS yang memiliki akses penuh.
- Izin eksplisit dalam kebijakan berbasis identitas atau berbasis sumber daya akan membatalkan pengaturan default ini.
- Jika ada batas izin, OrganizationsSCP, atau kebijakan sesi, mungkin akan mengganti izin dengan penolakan implisit.
- Penolakan secara tegas dalam kebijakan apa pun akan mengesampingkan izin apa pun.

Bagan alir berikut ini memberikan perincian tentang cara pengambilan keputusan. Diagram alir ini tidak mencakup dampak kebijakan berbasis sumber daya dan penolakan implisit dalam jenis kebijakan lainnya.



*A session principal is either a role session or an IAM federated user session.

1. Evaluasi penolakan – Secara default, semua permintaan ditolak. Ini disebut penolakan implisit. Bagian AWS kode penegakan mengevaluasi semua kebijakan dalam akun yang berlaku untuk permintaan. Ini termasuk AWS Organizations SCPs, kebijakan berbasis sumber daya, kebijakan berbasis identitas, batas izin, IAM dan kebijakan sesi. Dalam semua kebijakan tersebut, kode penegakan mencari pernyataan `Deny` yang berlaku untuk permintaan. Ini disebut penolakan secara tegas. Jika kode penegakan menemukan bahkan satu penolakan eksplisit yang berlaku, kode mengembalikan keputusan akhir Deny. Jika tidak ada penolakan eksplisit, evaluasi kode penegakan berlanjut.
2. Organizations SCPs — Kemudian kode penegakan mengevaluasi AWS Organizations kebijakan kontrol layanan (SCPs) yang berlaku untuk permintaan. SCPsberlaku untuk kepala sekolah akun tempat dilampirkan. SCPs Jika kode penegakan tidak menemukan `Allow` pernyataan yang berlaku di dalamSCPs, permintaan tersebut secara eksplisit ditolak, bahkan jika penolakan tersebut tersirat. Kode penegakan mengembalikan keputusan akhir Deny. Jika tidak adaSCP, atau jika SCP memungkinkan tindakan yang diminta, evaluasi kode penegakan berlanjut.
3. Kebijakan berbasis sumber daya — Dalam akun yang sama, kebijakan berbasis sumber daya berdampak pada evaluasi kebijakan secara berbeda tergantung pada jenis prinsipal yang mengakses sumber daya, dan prinsip yang diizinkan dalam kebijakan berbasis sumber daya. Bergantung pada jenis prinsipal, kebijakan berbasis sumber daya dapat menghasilkan keputusan

akhir`Allow`, bahkan jika ada penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi. `Allow`

Untuk sebagian besar sumber daya, Anda hanya memerlukan izin eksplisit untuk prinsipal baik dalam kebijakan berbasis identitas atau kebijakan berbasis sumber daya untuk memberikan akses. IAMkebijakan kepercayaan peran dan kebijakan KMS utama adalah pengecualian untuk logika ini, karena mereka harus secara eksplisit mengizinkan akses untuk prinsipal.

Logika kebijakan berbasis sumber daya berbeda dari jenis kebijakan lain jika prinsipal yang ditentukan adalah IAM pengguna, IAM peran, atau prinsipal sesi. Prinsipal sesi termasuk sesi IAM peran atau sesi pengguna IAM federasi. Jika kebijakan berbasis sumber daya memberikan izin langsung kepada IAM pengguna atau kepala sesi yang membuat permintaan, maka penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi tidak memengaruhi keputusan akhir.

Tabel berikut membantu Anda memahami dampak kebijakan berbasis sumber daya untuk tipe utama yang berbeda ketika penolakan implisit hadir dalam kebijakan berbasis identitas, batas izin, dan kebijakan sesi.

Tabel berikut menunjukkan kebijakan berbasis sumber daya dan penolakan implisit dalam jenis kebijakan lain untuk akun yang sama.

Prinsipal membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
IAMperan	Tidak berlaku	Tidak berlaku	Tidak berlaku	Tidak berlaku	Tidak berlaku	Peran itu sendiri tidak dapat membuat permintaan. Permintaan dibuat dengan sesi peran setelah peran diasumsikan.

Prinsipal membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
IAMsesi peran	Memungkinkan peran ARN atau Memungkinkan sesi peran ARN	Menyangkal secara implisit	Menyangkal secara implisit	Menyangkal secara implisit	DENY-Peran ARN atau ALLOW-Sesi peran ARN	Ketika prinsipal dalam kebijakan berbasis sumber daya berperan ARN, batas izin dan kebijakan sesi dievaluasi sebagai bagian dari keputusan akhir. Penyangkalan implisit dalam salah satu kebijakan menghasilkan DENY keputusan. Ketika prinsipal dalam kebijakan berbasis

Prinsipal membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
						sumber daya adalah sesi peran ARN, izin diberikan langsung ke sesi. Jenis kebijakan lain tidak mempengaruhi keputusan.
IAMpengguna	Memungkinkan IAM pengguna ARN	Menyangkal secara implisit	Menyangkal secara implisit	Tidak berlaku	ALLOW	Izin diberikan langsung kepada pengguna. Jenis kebijakan lain tidak mempengaruhi keputusan.

Prinsipal membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
IAMpengguna federasi () `GetFederationToken`	Memungkinkan IAM pengguna ARN atau Memungkinkan IAM sesi pengguna federasi ARN	Menyangkal secara implisit	Menyangkal secara implisit	Menyangkal secara implisit	DENY-IAM pengguna ARN atau ALLOW-IAM sesi pengguna federasi ARN	Jika prinsipal dalam kebijakan berbasis sumber daya adalah IAMpengguna ARN, penolakan implisit baik dalam batas izin atau kebijakan sesi menghasilkan a. DENY Ketika prinsipal dalam kebijakan berbasis sumber daya adalah sesi pengguna IAM

Prinsipal membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
						federasiARN, izin diberikan langsung ke sesi. Jenis kebijakan lain tidak mempengaruhi keputusan.

Prinsipal membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
pengguna root	Memungkinkan pengguna root ARN	Tidak berlaku	Tidak berlaku	Tidak berlaku	ALLOW	Pengguna root memiliki akses lengkap dan tidak terbatas ke semua sumber daya di Anda Akun AWS. Untuk mempelajari cara mengontrol akses ke pengguna root untuk akun di AWS Organizations, lihat Kebijakan kontrol layanan (SCPs) di Panduan Pengguna Organizations.

Prinsipal membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
AWS layanan utama	Memungkinkan sebuah AWS layanan utama	Tidak berlaku	Tidak berlaku	Tidak berlaku	ALLOW	Saat kebijakan berbasis sumber daya memberikan izin langsung ke AWS prinsip layanan, jenis kebijakan lainnya tidak mempengaruhi keputusan.

- IAMperan — Kebijakan berbasis sumber daya yang memberikan izin untuk IAM peran ARN dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi. Anda dapat menentukan peran ARN dalam elemen Principal atau kunci `aws:PrincipalArn` kondisi. Dalam kedua kasus, kepala sekolah yang membuat permintaan adalah sesi IAM peran.

  Batas izin dan kebijakan sesi tidak membatasi izin yang diberikan menggunakan kunci `aws:PrincipalArn` kondisi dengan wildcard (*) di elemen Principal, kecuali kebijakan berbasis identitas berisi penolakan eksplisit. Untuk informasi selengkapnya, lihat Prinsipal peran IAM.

  Contoh peran ARN

```
arn:aws:iam::111122223333:role/examplerole
```

- IAMsesi peran — Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin untuk sesi peran memberikan ARN izin langsung ke sesi IAM peran yang diasumsikan. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi. Ketika Anda mengambil peran dan membuat permintaan, kepala sekolah yang membuat permintaan adalah sesi IAM peran ARN dan bukan peran itu sendiri. ARN Untuk informasi selengkapnya, lihat Kepala sesi peran.

  Contoh sesi peran ARN

  ```
  arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
  ```

- IAMpengguna — Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada IAM pengguna ARN (yang bukan sesi pengguna gabungan) tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas atau batas izin.

  Contoh IAM pengguna ARN

  ```
  arn:aws:iam::111122223333:user/exampleuser
  ```

- IAMsesi pengguna federasi — Sesi pengguna IAM federasi adalah sesi yang dibuat dengan menelepon. `GetFederationToken` Ketika pengguna federasi membuat permintaan, prinsipal yang membuat permintaan adalah pengguna federasi ARN dan bukan IAM pengguna yang ARN berfederasi. Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada pengguna ARN federasi memberikan izin langsung ke sesi. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi.

  Namun, jika kebijakan berbasis sumber daya memberikan izin kepada pengguna yang melakukan ARN federasi, maka permintaan yang dibuat oleh IAM pengguna federasi selama sesi dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

  Contoh IAM sesi pengguna federasi ARN

  ```
  arn:aws:sts::111122223333:federated-user/exampleuser
  ```

4. Kebijakan berbasis identitas – Kemudian kode tersebut memeriksa kebijakan berbasis identitas untuk prinsipal. Untuk IAM pengguna, ini termasuk kebijakan dan kebijakan pengguna dari grup

tempat pengguna berada. Jika tidak ada kebijakan berbasis identitas atau tidak ada pernyataan dalam kebijakan berbasis identitas yang memungkinkan tindakan yang diminta, maka permintaan tersebut ditolak secara implisit dan kode mengembalikan keputusan akhir Deny. Jika ada pernyataan dalam kebijakan berbasis identitas yang berlaku yang memungkinkan tindakan yang diminta, kode akan berlanjut.

5. IAMbatas izin — Kode kemudian memeriksa apakah IAM entitas yang digunakan oleh prinsipal memiliki batas izin. Jika kebijakan yang digunakan untuk menetapkan batas izin tidak mengizinkan tindakan yang diminta, maka permintaan tersebut ditolak secara implisit. Kode tersebut memberikan keputusan akhir Tolak. Jika tidak ada batasan izin, atau jika batas izin memungkinkan tindakan yang diminta, kode berlanjut.
6. Kebijakan sesi — Kode kemudian memeriksa apakah prinsipal adalah prinsipal sesi. Prinsipal sesi termasuk sesi IAM peran atau sesi pengguna IAM federasi. Jika kepala sekolah bukan kepala sesi, kode penegakan mengembalikan keputusan akhir Izinkan.

   Untuk prinsipal sesi, kode memeriksa apakah kebijakan sesi diteruskan dalam permintaan. Anda dapat meneruskan kebijakan sesi saat menggunakan AWS CLI atau AWS APIuntuk mendapatkan kredensi sementara untuk peran atau pengguna IAM federasi.

   - Jika kebijakan sesi hadir dan tidak mengizinkan tindakan yang diminta, maka permintaan tersebut ditolak secara implisit. Kode tersebut memberikan keputusan akhir Tolak.
   - Jika tidak ada kebijakan sesi, kode memeriksa apakah prinsipal adalah sesi peran. Jika kepala sekolah adalah sesi peran, maka permintaan tersebut Diizinkan. Jika tidak, permintaan secara implisit ditolak dan kode mengembalikan keputusan akhir Deny.
   - Jika kebijakan sesi hadir dan memungkinkan tindakan yang diminta, maka kode penegakan mengembalikan keputusan akhir Izinkan.
7. Kesalahan — Jika AWS kode penegakan menemukan kesalahan pada titik mana pun selama evaluasi, kemudian menghasilkan pengecualian dan menutup.

Contoh evaluasi kebijakan berbasis identitas dan kebijakan berbasis sumber daya

Tipe kebijakan yang paling umum adalah kebijakan berbasis identitas dan kebijakan berbasis sumber daya. Ketika akses ke sumber daya diminta, AWS mengevaluasi semua izin yang diberikan oleh kebijakan untuk setidaknya satu Izinkan dalam akun yang sama. Penolakan eksplisit dalam salah satu kebijakan mengesampingkan izin.

> **Important**
>
> Jika kebijakan berbasis identitas atau kebijakan berbasis sumber daya dalam akun yang sama mengizinkan permintaan dan yang lainnya tidak, permintaan tetap diizinkan.

Asumsikan bahwa Carlos memiliki nama pengguna `carlossalazar` dan dia mencoba menyimpan file ke bucket Amazon S3 `amzn-s3-demo-bucket-carlossalazar-logs`.

Juga asumsikan bahwa kebijakan berikut dilampirkan ke `carlossalazar` IAM pengguna.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

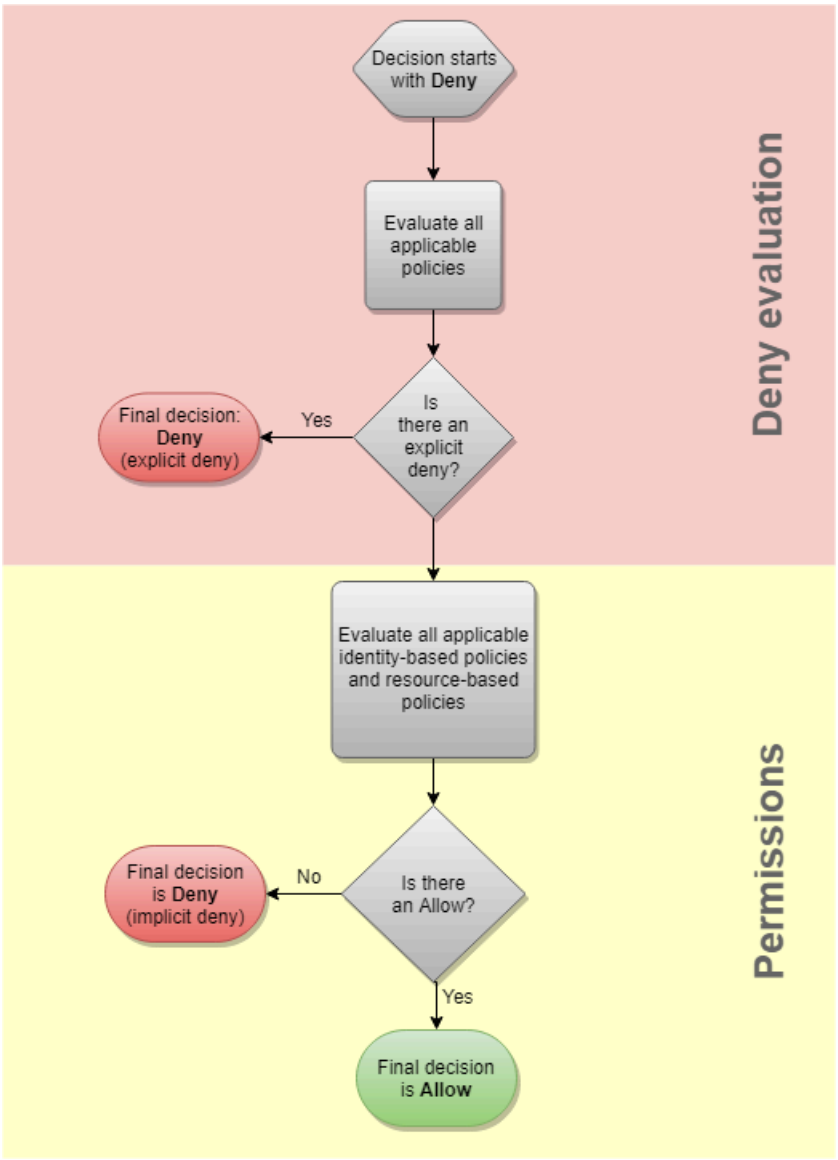
Pernyataan `AllowS3ListRead` dalam kebijakan ini memungkinkan Carlos melihat daftar semua bucket di akun. Pernyataan `AllowS3Self` memungkinkan Carlos mendapatkan akses penuh ke bucket dengan nama yang sama dengan nama penggunanya. Pernyataan `DenyS3Logs` menolak akses Carlos ke setiap bucket S3 mana pun dengan `log` dalam namanya.

Selain itu, kebijakan berbasis sumber daya berikut (disebut kebijakan buket) dilampirkan ke bucket `amzn-s3-demo-bucket-carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        }
    ]
}
```

Kebijakan ini menetapkan bahwa hanya pengguna `carlossalazar` yang dapat mengakses bucket `amzn-s3-demo-bucket-carlossalazar`.

Ketika Carlos membuat permintaannya untuk menyimpan file ke `amzn-s3-demo-bucket-carlossalazar-logs` ember, AWS menentukan kebijakan apa yang berlaku untuk permintaan tersebut. Dalam kasus ini, hanya kebijakan berbasis identitas dan kebijakan berbasis sumber daya yang berlaku. Keduanya adalah kebijakan izin. Karena batas izin tidak berlaku, logika evaluasi dikurangi ke logika berikut.



AWS pertama memeriksa `Deny` pernyataan yang berlaku untuk konteks permintaan. Ia menemukan satu, karena kebijakan berbasis identitas secara eksplisit menyangkal akses Carlos ke bucket S3 yang digunakan untuk logging. Akses Carlos ditolak.

Asumsikan bahwa dia kemudian menyadari kesalahannya dan mencoba menyimpan file tersebut ke bucket `amzn-s3-demo-bucket-carlossalazar`. AWS memeriksa `Deny` pernyataan dan tidak menemukannya. Kemudian memeriksa kebijakan izin. Baik kebijakan berbasis identitas maupun kebijakan berbasis sumber daya mengizinkan permintaan tersebut. Oleh karena itu, AWS memungkinkan permintaan. Jika salah satu dari mereka menolak pernyataan tersebut secara tegas, permintaan tersebut akan ditolak. Jika salah satu tipe kebijakan mengizinkan permintaan tersebut dan tipe yang lainnya tidak, permintaan tersebut masih diperbolehkan.

Perbedaan antara penolakan tegas dan implisit.

Permintaan menghasilkan penolakan eksplisit jika kebijakan yang berlaku mencakup pernyataan `Deny`. Jika kebijakan yang berlaku untuk permintaan mencakup pernyataan `Allow` dan `Deny`, pernyataan `Deny` mengalahkan pernyataan `Allow`. Permintaan ditolak secara tegas.

Penolakan implisit terjadi saat tidak ada pernyataan `Deny` yang berlaku, tetapi juga tidak ada pernyataan `Allow`. Karena IAM prinsipal ditolak akses secara default, mereka harus secara eksplisit diizinkan untuk melakukan tindakan. Jika tidak, akse pengguna akan ditolak secara implisit.

Saat Anda merancang strategi otorisasi, Anda harus membuat kebijakan dengan pernyataan `Allow` agar prinsipal Anda berhasil membuat permintaan. Namun, Anda dapat memilih kombinasi penyangkalan eksplisit dan implisit.

Misalnya, Anda dapat membuat kebijakan berikut yang mencakup tindakan yang diizinkan, tindakan yang ditolak secara implisit, dan tindakan yang ditolak secara eksplisit. `AllowGetList`Pernyataan ini memungkinkan akses hanya-baca ke IAM tindakan yang dimulai dengan awalan `Get` dan. `List` Semua tindakan lain diIAM, seperti `iam:CreatePolicy` secara implisit ditolak. `DenyReports`Pernyataan tersebut secara eksplisit menolak akses ke IAM laporan dengan menolak akses ke tindakan yang menyertakan `Report` akhiran, seperti. `iam:GetOrganizationsAccessReport` Jika seseorang menambahkan kebijakan lain ke prinsipal ini untuk memberi mereka akses ke IAM laporan, seperti`iam:GenerateCredentialReport`, permintaan terkait laporan masih ditolak karena penolakan eksplisit ini.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenyReports",
            "Effect": "Deny",
            "Action": "iam:*Report",
            "Resource": "*"
```

```
        }
    ]
}
```

Logika evaluasi kebijakan lintas akun

Anda dapat mengizinkan prinsipal di satu akun untuk mengakses sumber daya di akun kedua. Hal ini dikenal sebagai akses lintas akun. Saat Anda mengizinkan akses lintas akun, akun yang memiliki prinsipal disebut akun tepercaya. Akun yang memiliki sumber daya disebut akun kepercayaan.

Untuk memungkinkan akses lintas akun, Anda melampirkan kebijakan berbasis sumber daya ke sumber daya yang ingin Anda bagikan. Anda juga harus melampirkan kebijakan berbasis identitas pada identitas yang bertindak sebagai kepala sekolah dalam permintaan. Kebijakan berbasis sumber daya dalam akun kepercayaan harus menyebutkan prinsipal akun tepercaya yang akan memiliki akses ke sumber daya tersebut. Anda dapat menentukan seluruh akun atau penggunanya, IAM pengguna gabungan, IAM peran, atau sesi peran yang dianggap. Anda juga dapat menentukan AWS Pelayanan sebagai Principal Untuk informasi selengkapnya, lihat Cara menentukan kepala sekolah.

Kebijakan berbasis identitas prinsipal harus mengizinkan akses yang diminta ke sumber daya dalam layanan kepercayaan. Anda dapat melakukan ini dengan menentukan sumber daya atau dengan mengizinkan akses ke semua sumber daya (*). ARN

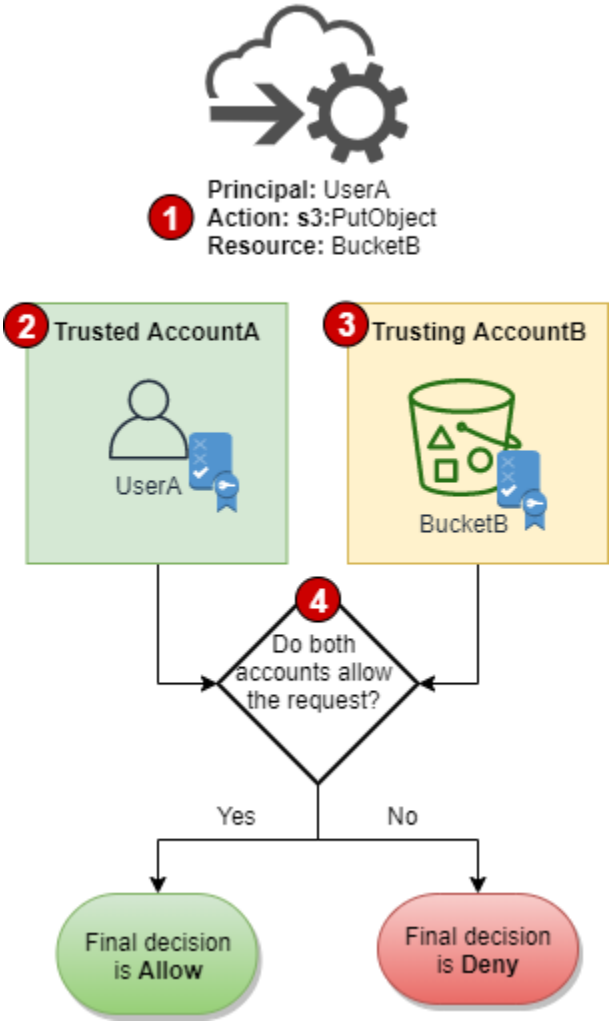
DiIAM, Anda dapat melampirkan kebijakan berbasis sumber daya ke IAM peran untuk mengizinkan kepala sekolah di akun lain mengambil peran tersebut. Kebijakan berbasis sumber daya peran ini disebut kebijakan kepercayaan peran. Setelah mengambil peran tersebut, prinsipal yang diizinkan dapat menggunakan kredensial sementara yang dihasilkan untuk mengakses beberapa sumber daya dalam akun Anda. Akses ini ditetapkan dalam kebijakan izin berbasis identitas dari peran tersebut. Untuk mempelajari cara memungkinkan akses lintas akun menggunakan peran berbeda dengan mengizinkan akses lintas akun menggunakan kebijakan berbasis sumber daya lainnya, lihat Akses sumber daya lintas akun di IAM.

> **Important**
>
> Layanan lain dapat memengaruhi logika evaluasi kebijakan. Misalnya, AWS Organizations mendukung kebijakan kontrol layanan yang dapat diterapkan pada prinsipal satu atau beberapa akun. AWS Resource Access Manager mendukung fragmen kebijakan yang mengontrol tindakan yang diizinkan oleh prinsipal untuk dilakukan pada sumber daya yang dibagikan dengannya.

Menentukan apakah permintaan lintas akun diizinkan atau ditolak.

Untuk permintaan lintas akun, pemohon dalam `AccountA` tepercaya harus memiliki kebijakan berbasis identitas. Kebijakan yang harus memungkinkan mereka membuat permintaan ke sumber daya dalam `AccountB` kepercayaan. Selain itu, kebijakan berbasis sumber daya di `AccountB` harus mengizinkan pemohon di `AccountA` untuk mengakses sumber daya.

Saat Anda membuat permintaan lintas akun, AWS melakukan dua evaluasi. AWS mengevaluasi permintaan di akun kepercayaan dan akun tepercaya. Untuk informasi selengkapnya tentang bagaimana permintaan dievaluasi di satu akun, lihat Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun.. Permintaan hanya diperbolehkan jika kedua evaluasi menghasilkan keputusan `Allow`.



1. Jika prinsipal di satu akun mengajukan permintaan untuk mengakses sumber daya di akun lain, ini adalah permintaan lintas akun.

2. Prinsipal mengajukan permohonan ada di akun tepercaya (`AccountA`). Saat AWS mengevaluasi akun ini, memeriksa kebijakan berbasis identitas dan kebijakan apa pun yang dapat membatasi kebijakan berbasis identitas. Untuk informasi selengkapnya, lihat Mengevaluasi kebijakan dalam satu akun.
3. Sumber daya yang diminta ada di akun kepercayaan (`AccountB`). Saat AWS mengevaluasi akun ini, memeriksa kebijakan berbasis sumber daya yang dilampirkan ke sumber daya yang diminta dan kebijakan apa pun yang dapat membatasi kebijakan berbasis sumber daya. Untuk informasi selengkapnya, lihat Mengevaluasi kebijakan dalam satu akun.
4. AWS mengizinkan permintaan hanya jika kedua evaluasi kebijakan akun mengizinkan permintaan tersebut.

Contoh evaluasi kebijakan lintas akun

Contoh berikut menunjukkan skenario di mana pengguna dalam satu akun diberikan izin oleh kebijakan berbasis sumber daya di akun kedua.

Asumsikan bahwa Carlos adalah pengembang dengan nama IAM pengguna `carlossalazar` di akun 11111111111111. Dia ingin menyimpan file ke bucket Amazon S3 `amzn-s3-demo-bucket-production-logs` di akun 222222222222.

Juga asumsikan bahwa kebijakan berikut dilampirkan ke `carlossalazar` IAM pengguna.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
```

```
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

Pernyataan `AllowS3ListRead` dalam kebijakan ini memungkinkan Carlos melihat daftar semua bucket di Amazon S3. Pernyataan `AllowS3ProductionObjectActions` memungkinkan Carlos sepenuhnya mengakses objek di bucket `amzn-s3-demo-bucket-production`. Pernyataan `DenyS3Logs` menolak akses Carlos ke setiap bucket S3 mana pun dengan `log` dalam namanya. Pernyataan ini juga menghalangi akses ke semua objek di bucket tersebut.

Selain itu, kebijakan berbasis sumber daya berikut (disebut kebijakan buket) diberlakukan untuk bucket `amzn-s3-demo-bucket-production` di akun 222222222222.

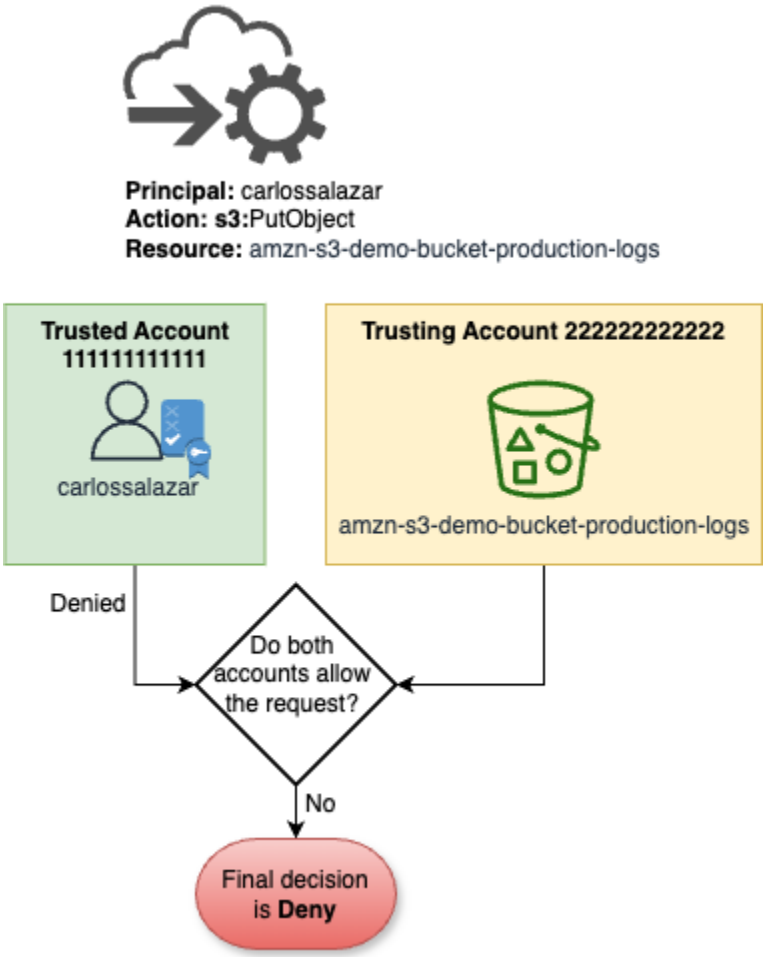
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:user/carlossalazar" },
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        }
    ]
}
```

Kebijakan ini memungkinkan pengguna `carlossalazar` untuk mengakses objek di bucket `amzn-s3-demo-bucket-production`. Dia dapat membuat dan mengedit, tetapi tidak menghapus objek dalam bucket. Dia tidak dapat mengelola bucket sendiri.

Ketika Carlos membuat permintaannya untuk menyimpan file ke `amzn-s3-demo-bucket-production-logs` ember, AWS menentukan kebijakan apa yang berlaku untuk permintaan

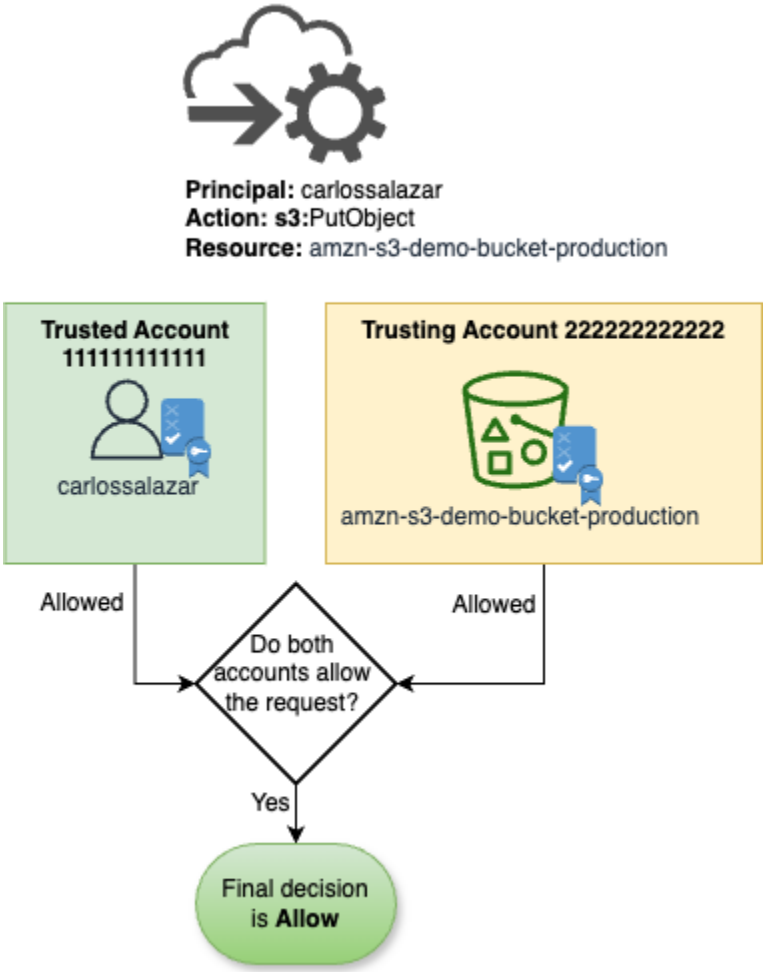
tersebut. Dalam hal ini, kebijakan berbasis identitas yang dilampirkan ke pengguna `carlossalazar` adalah satu-satunya kebijakan yang berlaku di akun `111111111111`. Di akun `222222222222`, tidak ada kebijakan berbasis sumber daya yang dilampirkan ke bucket `amzn-s3-demo-bucket-production-logs`. Saat AWS mengevaluasi akun`111111111111`, mengembalikan keputusan. `Deny` Ini karena pernyataan `DenyS3Logs` dalam kebijakan berbasis identitas secara eksplisit menolak akses ke bucket log. Untuk informasi selengkapnya tentang bagaimana permintaan dievaluasi di satu akun, lihat Menentukan apakah permintaan diizinkan atau ditolak dalamsebuah akun..

Karena permintaan tersebut secara tegas ditolak di dalam salah satu akun, keputusan akhir adalah menolak permintaan tersebut.



Asumsikan bahwa dia kemudian menyadari kesalahannya dan mencoba menyimpan file tersebut ke bucket `Production`. AWS pertama memeriksa akun `111111111111` untuk menentukan apakah permintaan diizinkan. Hanya kebijakan berbasis identitas yang berlaku, dan hal ini memungkinkan permintaan tersebut. AWS kemudian memeriksa akun`222222222222`. Hanya kebijakan berbasis sumber daya yang dilampirkan pada bucket `Production` yang berlaku, dan ini mengizinkan

permintaan. Karena kedua akun mengizinkan permintaan tersebut, keputusan akhir adalah untuk mengizinkan permintaan tersebut.



Tata bahasa IAM JSON kebijakan

Halaman ini menyajikan tata bahasa formal untuk bahasa yang digunakan untuk membuat JSON kebijakan diIAM. Kami menyajikan tata bahasa ini sehingga Anda dapat memahami cara menyusun dan memvalidasi kebijakan.

Untuk contoh kebijakan, lihat topik berikut:

- Kebijakan dan izin di AWS Identity and Access Management
- Contoh kebijakan berbasis identitas IAM
- Contoh Kebijakan untuk Bekerja di EC2 Konsol Amazon dan Contoh Kebijakan untuk Bekerja Dengan AWS CLI, Amazon EC2CLI, atau AWS SDKdi Panduan EC2 Pengguna Amazon.

- Contoh Kebijakan Bucket dan Contoh Kebijakan Pengguna di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Untuk contoh kebijakan yang digunakan di AWS layanan, pergi ke dokumentasi untuk layanan tersebut.

Topik

- Bahasa kebijakan dan JSON
- Konvensi yang digunakan dalam tata bahasa ini
- Tata Bahasa
- Catatan tata bahasa kebijakan

Bahasa kebijakan dan JSON

Kebijakan dinyatakan dalamJSON. Saat Anda membuat atau mengedit JSON kebijakan, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAMmengidentifikasi kesalahan JSON sintaks, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat Validasi kebijakan IAMAccess Analyzer.

Dalam dokumen ini, kami tidak memberikan deskripsi lengkap tentang apa yang dianggap validJSON. Namun, berikut adalah beberapa JSON aturan dasar:

- Spasi kosong antara entitas individu diperbolehkan.
- Nilai diapit dengan tanda petik. Tanda petik adalah bersifat opsional untuk nilai numerik dan Boolean.
- Banyak elemen (misalnya, `action_string_list` dan`resource_string_list`) dapat mengambil JSON array sebagai nilai. Susunan dapat mengambil satu atau beberapa nilai. Jika ada lebih dari satu nilai yang disertakan, susunan tersebut berada dalam kurung persegi (`[` dan `]`) dan dipisahkan dengan koma, seperti dalam contoh berikut:

  ```
  "Action" : ["ec2:Describe*","ec2:List*"]
  ```

- Tipe JSON data dasar (Boolean, number, dan string) didefinisikan dalam RFC7159.

Konvensi yang digunakan dalam tata bahasa ini

Konvensi berikut digunakan dalam tata bahasa ini:

- Karakter berikut adalah JSON token dan termasuk dalam kebijakan:

  ```
  { } [ ] " , :
  ```

- Karakter berikut adalah karakter khusus dalam tata bahasa dan tidak disertakan dalam kebijakan:

  ```
  = < > ( ) |
  ```

- Jika suatu elemen memungkinkan beberapa nilai, elemen tersebut ditunjukkan menggunakan nilai berulang, pemisah koma, dan elipsis (`...`). Contoh:

  ```
  [<action_string>, <action_string>, ...]
  ```

  ```
  <principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
  ```

  Jika beberapa nilai diperbolehkan, memasukkan hanya satu nilai juga akan valid. Hanya untuk satu nilai, koma di akhir harus dihilangkan. Jika elemen mengambil sebuah susunan (ditandai dengan [dan]) tetapi hanya satu nilai yang disertakan, tanda kurung bersifat opsional. Contoh:

  ```
  "Action": [<action_string>]
  ```

  ```
  "Action": <action_string>
  ```

- Tanda tanya (?) setelah elemen tertentu menunjukkan bahwa elemen tersebut bersifat opsional. Contoh:

  ```
  <version_block?>
  ```

  Namun, pastikan untuk merujuk pada catatan yang mengikuti daftar tata bahasa untuk perincian tentang elemen opsional.

- Garis vertikal (`|`) di antara elemen menunjukkan alternatif. Dalam tata bahasa, tanda kurung menentukan ruang lingkup alternatif. Contoh:

  ```
  ("Principal" | "NotPrincipal")
  ```

- Elemen yang harus berupa string literal diapit dengan tanda petik ganda (''). Contoh:

  ```
  <version_block> = "Version" : ("2008-10-17" | "2012-10-17")
  ```

Untuk catatan tambahan, lihat Catatan tata bahasa kebijakan mengikuti daftar tata bahasa.

Tata Bahasa

Daftar berikut menjelaskan tata bahasa kebijakan. Untuk konvensi yang digunakan dalam daftar, lihat bagian sebelumnya. Untuk informasi tambahan, lihat catatan setelahnya.

> **Note**
>
> Tata bahasa ini menjelaskan kebijakan yang ditandai dengan versi `2008-10-17` dan `2012-10-17`. Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan, di sisi lain, dibuat saat Anda membuat perubahan pada kebijakan yang dikelola pelanggan diIAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM buat versi baru dari kebijakan terkelola. Untuk mempelajari selengkapnya tentang elemen kebijakan `Version`, lihat IAMJSONelemen kebijakan: Version. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called “Kebijakan IAM versioning”.

```
policy  = {
     <version_block?>
     <id_block?>
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17" | "2012-10-17")

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
    <condition_block?>
}
```

```
<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

Catatan tata bahasa kebijakan

- Satu kebijakan dapat berisi serangkaian pernyataan.
- Kebijakan memiliki ukuran maksimum antara 2.048 karakter hingga 10.240 karakter, bergantung pada entitas yang menyertai kebijakan tersebut. Untuk informasi selengkapnya, lihat IAMdan AWS STS kuota. Perhitungan ukuran kebijakan tidak mencakup karakter spasi kosong.
- Elemen individu dilarang berisi beberapa kasus dari kunci yang sama. Misalnya, Anda tidak dapat menyertakan blok `Effect` dua kali dalam pernyataan yang sama.
- Blok dapat muncul dalam urutan apa pun. Misalnya, `version_block` dapat mengikuti `id_block` dalam kebijakan. Demikian pula, `effect_block`, `principal_block`, `action_block` dapat muncul dengan urutan apa pun dalam pernyataan.
- `id_block` bersifat opsional dalam kebijakan berbasis sumber daya. Komponen itu dilarang dimasukkan dalam kebijakan berbasis identitas.

- `principal_block`Elemen diperlukan dalam kebijakan berbasis sumber daya (misalnya, dalam kebijakan bucket Amazon S3) dan dalam kebijakan kepercayaan untuk peran. IAM Komponen itu dilarang dimasukkan dalam kebijakan berbasis identitas.
- Elemen `principal_map` di kebijakan bucket Amazon S3 dapat mencakup ID `CanonicalUser`. Sebagian besar kebijakan berbasis sumber daya tidak mendukung pemetaan ini. Untuk mempelajari lebih lanjut tentang menggunakan ID pengguna kanonik dalam kebijakan bucket, lihat Menentukan Prinsipal dalam Kebijakan di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.
- Setiap nilai string (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string`, dan versi string dari `condition_value`) dapat memiliki batasan panjang minimal dan maksimal, nilai tertentu yang diizinkan, atau format internal yang diperlukan.

Catatan tentang nilai string

Bagian ini menyediakan informasi tambahan tentang nilai string yang digunakan dalam elemen yang berbeda dalam kebijakan.

`action_string`

Terdiri dari namespace layanan, titik dua, dan nama tindakan Nama tindakan dapat mencakup kartu bebas. Contoh:

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
  "ec2:StopInstances"
]

"Action":"cloudformation:*"

"Action":"*"

"Action":[
  "s3:Get*",
  "s3:List*"
]
```

`policy_id_string`

Menyediakan cara untuk menyertakan informasi tentang kebijakan secara keseluruhan. Beberapa layanan, seperti Amazon SQS dan AmazonSNS, menggunakan `Id` elemen dengan cara yang dicadangkan. Kecuali jika dibatasi oleh layanan individu, policy_id_string dapat mencakup spasi. Beberapa layanan mengharuskan nilai ini menjadi unik dalam AWS akun.

> **Note**
>
> `id_block` diperbolehkan dalam kebijakan berbasis sumber daya, tetapi tidak dalam kebijakan berbasis identitas

Tidak ada batas panjang, meskipun string ini berkontribusi pada panjang kebijakan secara keseluruhan, yang dibatasi.

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

`sid_string`

Memberikan cara untuk menyertakan informasi tentang pernyataan individu. Untuk IAM kebijakan, karakter alfanumerik dasar (A-Z, a-z, 0-9) adalah satu-satunya karakter yang diizinkan dalam nilai. `Sid` Lainnya AWS Layanan yang mendukung kebijakan sumber daya mungkin memiliki persyaratan lain untuk `Sid` nilai tersebut. Misalnya, beberapa layanan mengharuskan nilai ini menjadi unik dalam Akun AWS, dan beberapa layanan memungkinkan karakter tambahan seperti spasi dalam `Sid` nilai.

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

`principal_id_string`

Menyediakan cara untuk menentukan prinsipal menggunakan Amazon Resource Name (ARN) Akun AWS, IAM pengguna, IAM peran, pengguna federasi, atau pengguna peran yang dianggap. Untuk Akun AWS, Anda juga dapat menggunakan formulir pendek `AWS:`*`accountnumber`* alih-alih penuhARN. Untuk semua opsi termasuk AWS layanan, peran yang diasumsikan, dan sebagainya, lihatCara menentukan kepala sekolah.

Perhatikan bahwa Anda dapat menggunakan * hanya untuk menentukan "semua orang/anonim". Anda tidak dapat menggunakannya untuk menentukan bagian dari nama atauARN.

`resource_string`

Dalam kebanyakan kasus, terdiri dari Amazon Resource Name (ARN).

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::amzn-s3-demo-bucket/*"
```

`condition_type_string`

Mengidentifikasi tipe kondisi yang diuji, seperti `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanEquals`, `Bool`, `BinaryEquals`, `IpAddress`, `ArnEquals`, dll. Untuk daftar lengkap tipe kondisi, lihat IAMJSONelemen kebijakan: Operator kondisi.

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}

"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

`condition_key_string`

Mengidentifikasi kunci kondisi yang nilainya akan diuji untuk menentukan apakah kondisi terpenuhi. AWS mendefinisikan satu set kunci kondisi yang tersedia di semua AWS layanan, termasuk`aws:PrincipalType`,`aws:SecureTransport`, dan`aws:userid`.

Untuk daftar AWS tombol kondisi, lihatAWS kunci konteks kondisi global. Untuk kunci kondisi kepatuhan yang bersifat khusus terhadap layanan, lihat dokumentasi untuk layanan tersebut seperti berikut:

- Menentukan Ketentuan dalam Kebijakan di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon
- IAMKebijakan untuk Amazon EC2 di Panduan EC2 Pengguna Amazon.

```
"Condition":{
  "Bool": {
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

`condition_value_string`

Mengidentifikasi nilai condition_key_string yang menentukan apakah kondisi terpenuhi. Untuk daftar lengkap nilai valid untuk jenis kondisi, lihatIAMJSONelemen kebijakan: Operator kondisi.

```
"Condition":{
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
   "ID",
   "PostDateTime"
         ]
  }
}
```

AWS kebijakan terkelola untuk fungsi pekerjaan

Sebaiknya gunakan kebijakan yang memberikan hak istimewa paling sedikit, atau hanya memberikan izin yang diperlukan untuk melakukan tugas. Cara paling aman untuk memberikan hak istimewa paling sedikit adalah dengan menulis kebijakan khusus hanya dengan izin yang diperlukan oleh tim Anda. Anda harus membuat proses untuk memungkinkan tim Anda meminta lebih banyak izin bila diperlukan. Butuh waktu dan keahlian untuk membuat kebijakan terkelola IAM pelanggan yang hanya memberi tim Anda izin yang mereka butuhkan.

Untuk mulai menambahkan izin ke IAM identitas Anda (pengguna, grup pengguna, dan peran), Anda dapat menggunakannya. AWS kebijakan terkelola AWS kebijakan terkelola mencakup kasus penggunaan umum dan tersedia di Anda Akun AWS. AWS kebijakan terkelola tidak memberikan izin hak istimewa paling sedikit. Anda harus mempertimbangkan risiko keamanan memberikan izin kepada kepala sekolah Anda lebih banyak daripada yang mereka butuhkan untuk melakukan pekerjaan mereka.

Anda dapat melampirkan kebijakan AWS terkelola, termasuk fungsi pekerjaan, ke IAM identitas apa pun. Untuk beralih ke izin hak istimewa terkecil, Anda dapat menjalankan AWS Identity and Access Management Access Analyzer untuk memantau prinsipal dengan kebijakan terkelola. AWS Setelah mengetahui izin yang mereka gunakan, Anda dapat menulis kebijakan khusus atau membuat kebijakan hanya dengan izin yang diperlukan untuk tim Anda. Ini kurang aman, tetapi memberikan lebih banyak fleksibilitas saat Anda mempelajari bagaimana tim Anda menggunakan AWS.

AWS kebijakan terkelola untuk fungsi pekerjaan dirancang untuk menyelaraskan secara dekat dengan fungsi pekerjaan umum di industri TI. Anda dapat menggunakan kebijakan ini untuk memberikan izin yang diperlukan untuk melaksanakan tugas yang diharapkan dari seseorang dalam fungsi pekerjaan tertentu. Kebijakan ini mengonsolidasikan izin untuk banyak layanan ke dalam kebijakan tunggal yang lebih mudah digunakan daripada memiliki izin yang tersebar di banyak kebijakan.

Menggunakan Peran untuk Menggabungkan Layanan

Beberapa kebijakan menggunakan peran IAM layanan untuk membantu Anda memanfaatkan fitur yang ditemukan di AWS layanan lain. Kebijakan ini memberikan akses `keiam:passrole`, yang memungkinkan pengguna dengan kebijakan untuk meneruskan peran ke AWS layanan. Peran ini mendelegasikan IAM izin ke AWS layanan untuk melakukan tindakan atas nama Anda.

Anda harus membuat peran sesuai dengan kebutuhan Anda. Misalnya, kebijakan Administrator Jaringan memungkinkan pengguna dengan kebijakan untuk meneruskan peran bernama "flow-logs-

vpc" ke CloudWatch layanan Amazon. CloudWatch menggunakan peran itu untuk mencatat dan menangkap lalu lintas IP yang VPCs dibuat oleh pengguna.

Untuk mengikuti praktik terbaik keamanan, kebijakan untuk fungsi pekerjaan mencakup filter yang membatasi nama peran valid yang dapat diberikan. Tindakan ini membantu menghindari memberikan izin yang tidak perlu. Jika pengguna Anda memang memerlukan peran layanan opsional, Anda harus membuat peran yang mengikuti konvensi penamaan yang ditetapkan dalam kebijakan. Kemudian, Anda memberikan izin untuk peran tersebut. Setelah selesai, pengguna dapat mengonfigurasi layanan untuk menggunakan peran, memberikan izin apa pun yang diberikan oleh peran.

Di bagian berikut, nama setiap kebijakan adalah tautan ke halaman perincian kebijakan di AWS Management Console. Di sana Anda dapat melihat dokumen kebijakan dan meninjau izin yang diberikan.

Fungsi pekerjaan administrator

AWS nama kebijakan terkelola: AdministratorAccess

Kasus penggunaan: Pengguna ini memiliki akses penuh dan dapat mendelegasikan izin untuk setiap layanan dan sumber daya di AWS.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan semua tindakan untuk semua AWS layanan dan untuk semua sumber daya di akun. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat AdministratorAccessdi Panduan Referensi Kebijakan AWS Terkelola.

> **Note**
>
> Sebelum IAM pengguna atau peran dapat mengakses AWS Billing and Cost Management konsol dengan izin dalam kebijakan ini, Anda harus terlebih dahulu mengaktifkan akses IAM pengguna dan peran. Untuk melakukannya, ikuti petunjuk di Berikan akses ke konsol penagihan untuk mendelegasikan akses ke konsol penagihan.

Fungsi pekerjaan penagihan

AWS nama kebijakan terkelola: Penagihan

Kasus penggunaan: Pengguna ini perlu melihat informasi penagihan, menyiapkan pembayaran, dan mengotorisasi pembayaran. Pengguna dapat memantau biaya yang terakumulasi untuk seluruh AWS layanan.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi tugas, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin penuh untuk mengelola penagihan, biaya, metode pembayaran, anggaran, dan laporan. Untuk contoh kebijakan manajemen biaya tambahan, lihat contoh AWS Billing kebijakan di Panduan AWS Billing and Cost Management Pengguna. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat Penagihan di Panduan Referensi Kebijakan AWS Terkelola.

> **Note**
>
> Sebelum IAM pengguna atau peran dapat mengakses AWS Billing and Cost Management konsol dengan izin dalam kebijakan ini, Anda harus terlebih dahulu mengaktifkan akses IAM pengguna dan peran. Untuk melakukannya, ikuti petunjuk di Berikan akses ke konsol penagihan untuk mendelegasikan akses ke konsol penagihan.

Fungsi pekerjaan administrator basis data

AWS nama kebijakan terkelola: DatabaseAdministrator

Kasus penggunaan: Pengguna ini menyiapkan, mengonfigurasi, dan memelihara database di Cloud. AWS

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi tugas, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat, mengonfigurasi, dan memelihara basis data. Ini termasuk akses ke layanan AWS database, seperti Amazon DynamoDB, Amazon Relational Database RDS Service (), dan Amazon Redshift. Lihat kebijakan untuk mengetahui daftar lengkap layanan basis data yang mendukung kebijakan ini. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat DatabaseAdministratordi Panduan Referensi Kebijakan AWS Terkelola.

Kebijakan fungsi pekerjaan ini mendukung kemampuan untuk meneruskan peran ke AWS layanan. Kebijakan ini mengizinkan tindakan `iam:PassRole` hanya untuk peran yang disebutkan dalam tabel berikut. Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	Pilih kebijakan AWS terkelola ini
Memungkinkan pengguna untuk memantau RDS database	rds-monitoring-role	RDSPeran Amazon untuk Pemantauan yang Ditingkatkan	A mazonRDSEnhanced MonitoringRole
Memungkinkan AWS Lambda untuk memantau database Anda dan mengakses database eksternal	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess
Izinkan Lambda mengunggah file ke Amazon S3 dan ke cluster Amazon Redshift dengan DynamoDB	lambda_exec_role	AWS Lambda	Membuat kebijakan pengelolaan baru sebagaimana yang ditentukan dalam Blog Big Data AWS
Izinkan fungsi Lambda bertindak sebagai pemicu untuk tabel DynamoDB Anda	lambda-dinamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	Pilih kebijakan AWS terkelola ini
Izinkan fungsi Lambda mengakses Amazon RDS VPC	lambda-vpc-execution-role	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Lambda Pengembang	AWSLambda VPCAccessExecutionRole
Izinkan AWS Data Pipeline untuk mengakses AWS sumber daya Anda	DataPipelineDefaultRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Data Pipeline Pengembang	AWS Data Pipeline Dokumentasi mencantumkan izin yang diperlukan untuk kasus penggunaan ini. Lihat IAMperan untuk AWS Data Pipeline
Izinkan aplikasi Anda berjalan di EC2 instans Amazon untuk mengakses sumber daya Anda AWS	DataPipelineDefaultResourceRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Data Pipeline Pengembang	Amazon EC2RoleforDataPipelineRole

Fungsi pekerjaan ilmuwan data

AWS nama kebijakan terkelola: DataScientist

Kasus penggunaan: Pengguna ini menjalankan pekerjaan dan kueri Hadoop. Pengguna juga mengakses dan menganalisis informasi untuk analitik data dan kecerdasan bisnis.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat, mengelola, dan menjalankan kueri di EMR klaster Amazon dan melakukan analisis data dengan alat seperti Amazon. QuickSight Kebijakan ini mencakup akses ke layanan ilmuwan data tambahan, seperti AWS Data Pipeline, AmazonEC2, Amazon Kinesis, Amazon Machine Learning, dan. SageMaker Lihat kebijakan untuk mengetahui daftar lengkap layanan ilmuwan data yang mendukung kebijakan ini. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat DataScientistdi Panduan Referensi Kebijakan AWS Terkelola.

Kebijakan fungsi pekerjaan ini mendukung kemampuan untuk meneruskan peran ke AWS layanan. Satu pernyataan memungkinkan meneruskan peran apa pun ke SageMaker. Pernyataan lainnya mengizinkan tindakan `iam:PassRole` hanya untuk peran yang disebutkan dalam tabel berikut. Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
Izinkan EC2 instans Amazon mengakses layanan dan sumber daya yang sesuai untuk klaster	EMR-EC2_DefaultRole	Amazon EMR untuk EC2	AmazonElasticMapReduceforEC2Role
Izinkan EMR akses Amazon untuk mengakses EC2 layanan Amazon dan sumber daya untuk cluster	EMR_DefaultRole	Amazon EMR	SebuahmazonEMRServicePolicy_v2

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
Izinkan Kinesis Managed Service untuk Apache Flink untuk mengakses sumber data streaming	kinesis-*	Membuat peran dengan kebijakan kepercayaan sebagaimana yang ditentukan dalam Blog Big Data AWS	Lihat Blog Big Data AWS yang menguraikan empat kemungkinan opsi, bergantung pada kasus penggunaan Anda
Izinkan AWS Data Pipeline untuk mengakses AWS sumber daya Anda	DataPipelineDefaultRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Data Pipeline Pengembang	AWS Data Pipeline Dokumentasi mencantumkan izin yang diperlukan untuk kasus penggunaan ini. Lihat IAMperan untuk AWS Data Pipeline
Izinkan aplikasi Anda berjalan di EC2 instans Amazon untuk mengakses sumber daya Anda AWS	DataPipelineDefaultResourceRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Data Pipeline Pengembang	Amazon EC2RoleforDataPipelineRole

Fungsi pekerjaan pengguna daya developer

AWS nama kebijakan terkelola: PowerUserAccess

Kasus penggunaan: Pengguna ini melakukan tugas pengembangan aplikasi dan dapat membuat dan mengonfigurasi sumber daya dan layanan yang mendukung pengembangan aplikasi AWS sadar.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Pernyataan pertama kebijakan ini menggunakan `NotAction`elemen untuk mengizinkan semua tindakan untuk semua AWS layanan dan untuk semua sumber daya kecuali AWS Identity and Access Management, AWS Organizations, dan AWS Account Management. Pernyataan kedua memberikan IAM izin untuk membuat peran terkait layanan. Ini diwajibkan oleh beberapa layanan yang harus mengakses sumber daya di layanan lain, seperti bucket Amazon S3. Ini juga memberikan izin kepada Organizations untuk melihat informasi tentang organisasi pengguna, termasuk email akun manajemen dan batasan organisasi. Meskipun kebijakan ini membatasiIAM, Organizations, kebijakan ini memungkinkan pengguna untuk melakukan semua tindakan Pusat IAM IAM Identitas jika Pusat Identitas diaktifkan. Ini juga memberikan izin Manajemen Akun untuk melihat AWS Wilayah mana yang diaktifkan atau dinonaktifkan untuk akun tersebut.

Fungsi pekerjaan administrator jaringan

AWS nama kebijakan terkelola: NetworkAdministrator

Kasus penggunaan: Pengguna ini ditugaskan untuk menyiapkan dan memelihara sumber daya AWS jaringan.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memelihara sumber daya jaringan di Auto Scaling, EC2 Amazon,, Route 53, AWS Direct Connect Amazon, Elastic Load Balancing, CloudFront Amazon,, CloudWatch Log AWS Elastic Beanstalk, SNS CloudWatch Amazon S3,, dan Amazon Virtual Private IAM Cloud. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat NetworkAdministratordi Panduan Referensi Kebijakan AWS Terkelola.

Fungsi pekerjaan ini membutuhkan kemampuan untuk meneruskan peran ke AWS layanan. Kebijakan ini memberikan `iam:GetRole` dan `iam:PassRole` hanya untuk peran yang ditentukan dalam tabel berikut. Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
VPCMemungkinkan Amazon membuat dan mengelola CloudWatch log di Log atas nama pengguna untuk memantau lalu lintas IP yang masuk dan keluar VPC	aliran-log-*	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan VPC Pengguna Amazon	Kasus penggunaan ini tidak memiliki kebijakan AWS terkelola yang ada, tetapi dokumentasi mencantumkan izin yang diperlukan. Lihat Panduan VPC Pengguna Amazon.

Akses hanya-baca

AWS nama kebijakan terkelola: ReadOnlyAccess

Kasus penggunaan: Pengguna ini memerlukan akses hanya-baca ke setiap sumber daya dalam file. Akun AWS

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi tugas, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan:Kebijakan ini memberikan izin untuk daftar, mendapatkan, menguraikan, dan sebaliknya melihat sumber daya dan atribut mereka. Ini tidak termasuk fungsi bermutasi seperti membuat atau menghapus. Kebijakan ini mencakup akses hanya-baca ke AWS layanan terkait keamanan, seperti dan. AWS Identity and Access Management AWS Billing and Cost Management Lihat kebijakan untuk daftar lengkap layanan dan tindakan yang didukung kebijakan ini.

Fungsi pekerjaan auditor keamanan

AWS nama kebijakan terkelola: SecurityAudit

Kasus penggunaan: Pengguna ini memantau akun agar mematuhi persyaratan keamanan. Pengguna ini dapat mengakses catatan dan peristiwa untuk menginvestigasi kemungkinan adanya pelanggaran keamanan atau kemungkinan adanya aktivitas berbahaya.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk melihat data konfigurasi untuk banyak AWS layanan dan meninjau log mereka. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat SecurityAuditdi Panduan Referensi Kebijakan AWS Terkelola.

Fungsi pekerjaan pengguna

AWS nama kebijakan terkelola: SupportUser

Kasus penggunaan: Pengguna ini menghubungi AWS Support, membuat kasus dukungan, dan melihat status kasus yang ada.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memperbarui AWS Support kasus. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat SupportUserdi Panduan Referensi Kebijakan AWS Terkelola.

Fungsi pekerjaan administrator sistem

AWS nama kebijakan terkelola: SystemAdministrator

Kasus penggunaan: Pengguna ini mengatur dan memelihara sumber daya untuk operasi pengembangan.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memelihara sumber daya di berbagai AWS layanan, termasuk, Amazon,,, CloudWatch, AWS CodeCommit, AWS CloudTrail Amazon AWS CodeDeploy, AWS Config,,EC2, AWS Directory Service AWS Lambda Amazon AWS Identity and Access Management AWS Key Management Service, Route 53RDS, Amazon S3, Amazon, SES Amazon, dan SQS AWS Trusted Advisor Amazon. VPC Untuk informasi selengkapnya tentang kebijakan terkelola, lihat SystemAdministratordi Panduan Referensi Kebijakan AWS Terkelola.

Fungsi pekerjaan ini membutuhkan kemampuan untuk meneruskan peran ke AWS layanan. Kebijakan ini memberikan `iam:GetRole` dan `iam:PassRole` hanya untuk peran yang ditentukan dalam tabel berikut. Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
Izinkan aplikasi yang berjalan dalam EC2 instance di ECS klaster Amazon untuk mengakses Amazon ECS	ecr-sysadmin-*	EC2Peran Amazon untuk Layanan EC2 Kontainer	Amazon EC2ContainerServiceforEC2Role
Memungkinkan pengguna untuk memantau basis data	rds-monitoring-role	RDSPeran Amazon untuk Pemantauan yang Ditingkatkan	A mazonRDSEnhancedMonitoringRole
Izinkan aplikasi yang berjalan dalam EC2 instance untuk mengakses AWS sumber daya.	ec2-sysadmin-*	Amazon EC2	Contoh kebijakan untuk peran yang memberikan akses ke bucket S3 seperti yang ditunjukkan dalam Panduan EC2 Pengguna Amazon; sesuaikan sesuai kebutuhan

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
Izinkan Lambda membaca aliran DynamoDB dan menulis ke Log CloudWatch	lambda-sysadmin-*	AWS Lambda	AWSLambda DynamoDBE xecutionRole

Fungsi tugas pengguna hanya lihat

AWS nama kebijakan terkelola: ViewOnlyAccess

Kasus penggunaan: Pengguna ini dapat melihat daftar AWS sumber daya dan metadata dasar di akun di seluruh layanan. Pengguna tidak dapat membaca konten sumber daya atau metadata yang melampaui kuota dan informasi daftar sumber daya.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di IAM konsol, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan`List*`,,, `Describe*` `Get*` `View*`, dan `Lookup*` akses ke sumber daya untuk AWS layanan. Untuk melihat tindakan apa yang termasuk dalam kebijakan ini untuk setiap layanan, lihat ViewOnlyAccess. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat ViewOnlyAccessdi Panduan Referensi Kebijakan AWS Terkelola.

Pembaruan kebijakan AWS terkelola untuk fungsi pekerjaan

Semua kebijakan ini dikelola oleh AWS dan terus diperbarui untuk menyertakan dukungan untuk layanan baru dan kemampuan baru saat ditambahkan oleh AWS layanan. Kebijakan ini tidak dapat dimodifikasi oleh pelanggan. Anda dapat membuat salinan kebijakan dan kemudian memodifikasi salinannya, tetapi salinan itu tidak diperbarui secara otomatis karena AWS memperkenalkan layanan dan API operasi baru.

Untuk kebijakan fungsi pekerjaan, Anda dapat melihat riwayat versi serta waktu dan tanggal setiap pembaruan di IAM konsol. Untuk melakukannya, gunakan tautan di halaman ini untuk melihat detail kebijakan. Lalu pilih tab Versi kebijakan untuk melihat versi. Halaman ini menunjukkan 25 versi terakhir dari sebuah kebijakan. Untuk melihat semua versi kebijakan, panggil get-policy-version AWS CLI perintah atau GetPolicyVersionAPIoperasi.

> **Note**
>
> Anda dapat memiliki hingga lima versi kebijakan yang dikelola pelanggan, tetapi AWS mempertahankan riwayat versi lengkap kebijakan AWS terkelola.

Menciptakan peran dan memberlakukan kebijakan (konsol)

Beberapa kebijakan yang tercantum sebelumnya memberikan kemampuan untuk mengonfigurasi AWS layanan dengan peran yang memungkinkan layanan tersebut melakukan operasi atas nama Anda. Kebijakan fungsi pekerjaan menentukan nama peran yang tepat yang harus Anda gunakan atau setidaknya menyertakan awalan yang menentukan bagian pertama dari nama yang dapat digunakan. Untuk membuat salah satu peran ini, lakukan langkah-langkah dalam prosedur berikut.

Untuk membuat peran untuk Layanan AWS (IAMkonsol)

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi IAM konsol, pilih Peran, lalu pilih Buat peran.
3. Untuk jenis entitas Tepercaya, pilih Layanan AWS.
4. Untuk kasus Layanan atau penggunaan, pilih layanan, lalu pilih kasus penggunaan. Kasus penggunaan ditentukan oleh layanan untuk menyertakan kebijakan kepercayaan yang diperlukan layanan.
5. Pilih Berikutnya.
6. Untuk kebijakan Izin, opsi bergantung pada kasus penggunaan yang Anda pilih:
   - Jika layanan menentukan izin untuk peran tersebut, Anda tidak dapat memilih kebijakan izin.
   - Pilih dari serangkaian kebijakan izin terbatas.
   - Pilih dari semua kebijakan izin.
   - Pilih kebijakan tanpa izin, buat kebijakan setelah peran dibuat, lalu lampirkan kebijakan ke peran.
7. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.
   a. Buka bagian Setel batas izin, lalu pilih Gunakan batas izin untuk mengontrol izin peran maksimum.

IAMmenyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda.

b. Pilih kebijakan yang akan digunakan untuk batas izin.

8. Pilih Berikutnya.
9. Untuk nama Peran, opsi bergantung pada layanan:

   - Jika layanan menentukan nama peran, Anda tidak dapat mengedit nama peran.
   - Jika layanan mendefinisikan awalan untuk nama peran, Anda dapat memasukkan akhiran opsional.
   - Jika layanan tidak menentukan nama peran, Anda dapat memberi nama peran.

   > **Important**
   >
   > Saat Anda memberi nama peran, perhatikan hal berikut:
   >
   > - Nama peran harus unik di dalam diri Anda Akun AWS, dan tidak dapat dibuat unik berdasarkan kasus.
   >
   >   Misalnya, jangan membuat peran bernama keduanya **`PRODROLE`** dan**`prodrole`**. Ketika nama peran digunakan dalam kebijakan atau sebagai bagian dariARN, nama peran akan peka huruf besar/kecil, namun ketika nama peran muncul kepada pelanggan di konsol, seperti selama proses login, nama peran tersebut tidak peka huruf besar/kecil.
   > - Anda tidak dapat mengedit nama peran setelah dibuat karena entitas lain mungkin mereferensikan peran tersebut.

10. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran tersebut.
11. (Opsional) Untuk mengedit kasus penggunaan dan izin untuk peran, di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin, pilih Edit.
12. (Opsional) Untuk membantu mengidentifikasi, mengatur, atau mencari peran, tambahkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tagIAM, lihat Menandai IAM sumber daya di Panduan IAM Pengguna.
13. Tinjau peran lalu pilih Buat peran.

Contoh 1: Mengonfigurasi pengguna sebagai administrator basis data (konsol)

Contoh ini menunjukkan langkah-langkah yang diperlukan untuk mengkonfigurasi Alice, IAM pengguna, sebagai Administrator Database. Anda menggunakan informasi di baris pertama tabel di bagian itu dan memungkinkan pengguna untuk mengaktifkan RDS pemantauan Amazon. Anda melampirkan DatabaseAdministratorkebijakan ke IAM pengguna Alice sehingga mereka dapat mengelola layanan database Amazon. Kebijakan itu juga memungkinkan Alice untuk meneruskan peran yang dipanggil `rds-monitoring-role` ke RDS layanan Amazon yang memungkinkan layanan untuk memantau RDS database Amazon atas nama mereka.

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Pilih Kebijakan, **`database`** ketik kotak pencarian, lalu tekan enter.
3. Pilih tombol radio untuk DatabaseAdministratorkebijakan, pilih Tindakan, lalu pilih Lampirkan.
4. Dalam daftar entitas, pilih Alice dan kemudian pilih Lampirkan kebijakan. Alice sekarang dapat mengelola database AWS . Namun, agar Alice dapat memantau basis data tersebut, Anda harus mengonfigurasi peran layanan.
5. Di panel navigasi IAM konsol, pilih Peran, lalu pilih Buat peran.
6. Pilih jenis peran AWS Layanan, lalu pilih Amazon RDS.
7. Pilih RDSPeran Amazon untuk kasus penggunaan Pemantauan yang Ditingkatkan.
8. Amazon RDS mendefinisikan izin untuk peran Anda. Pilih Next: Review (Berikutnya: Tinjauan) untuk melanjutkan.
9. Nama peran harus salah satu yang ditentukan oleh DatabaseAdministrator kebijakan yang dimiliki Alice sekarang. Salah satunya adalah **`rds-monitoring-role`**. Masukkan itu untuk nama Peran.
10. (Opsional) Untuk Deskripsi peran, masukkan deskripsi.
11. Setelah meninjau perinciannya, pilih Buat peran.
12. Alice sekarang dapat mengaktifkan RDSEnhanced Monitoring di bagian Monitoring RDS konsol Amazon. Misalnya, mereka mungkin melakukan ini ketika mereka membuat instance DB, membuat replika baca, atau memodifikasi instance DB. Mereka harus memasukkan nama peran yang mereka buat (rds-monitoring-role) di kotak Peran Pemantauan saat mereka menyetel Aktifkan Pemantauan yang Ditingkatkan ke Ya.

Contoh 2: Mengonfigurasi pengguna sebagai administrator jaringan (konsol)

Contoh ini menunjukkan langkah-langkah yang diperlukan untuk mengkonfigurasi Jorge, IAM pengguna, sebagai Administrator Jaringan. Ini menggunakan informasi dalam tabel di bagian itu untuk memungkinkan Jorge memantau lalu lintas IP yang pergi ke dan dari aVPC. Ini juga memungkinkan Jorge untuk menangkap informasi itu di log di CloudWatch Log. Anda melampirkan NetworkAdministratorkebijakan ke IAM pengguna Jorge sehingga mereka dapat mengonfigurasi sumber daya AWS jaringan. Kebijakan itu juga memungkinkan Jorge untuk meneruskan peran yang namanya dimulai `flow-logs*` ke Amazon EC2 saat Anda membuat log alur. Dalam skenario ini, tidak seperti Contoh 1, tidak ada jenis peran layanan yang ditentukan sebelumnya sehingga Anda harus melakukan beberapa langkah secara berbeda.

1. Masuk ke AWS Management Console dan buka IAM konsol di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan lalu masukkan **`network`** di kotak pencarian, lalu tekan enter.
3. Pilih tombol radio di sebelah NetworkAdministratorkebijakan, pilih Tindakan, lalu pilih Lampirkan.
4. Dalam daftar pengguna, pilih kotak centang di sebelah Jorge, lalu pilih Lampirkan kebijakan. Jorge sekarang dapat mengelola sumber daya AWS jaringan. Namun, untuk mengaktifkan pemantauan lalu lintas IP di AndaVPC, Anda harus mengonfigurasi peran layanan.
5. Karena peran layanan yang perlu Anda buat tidak memiliki kebijakan yang dikelola sebelumnya, Anda harus membuatnya terlebih dahulu. Pada panel navigasi, pilih Kebijakan, lalu pilih Buat kebijakan.
6. Di bagian Editor kebijakan, pilih JSONopsi dan salin teks dari dokumen JSON kebijakan berikut. Tempelkan teks ini ke dalam kotak JSONteks.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
```

```
  ]
}
```

7. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

> **Note**
>
> Anda dapat beralih antara opsi Visual dan JSONeditor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

8. Pada halaman Tinjau dan buat, ketik **`vpc-flow-logs-policy-for-service-role`** nama kebijakan. Tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul di daftar kebijakan terkelola dan siap dilampirkan.
9. Di panel navigasi IAM konsol, pilih Peran, lalu pilih Buat peran.
10. Pilih jenis peran AWS Layanan, lalu pilih Amazon EC2.
11. Pilih kasus EC2 penggunaan Amazon.
12. Pada halaman Lampirkan kebijakan izin, pilih kebijakan yang Anda buat sebelumnya, vpc-flow-logs-policy- for-service-role, lalu pilih Berikutnya: Tinjau.
13. Nama peran harus diizinkan oleh NetworkAdministrator kebijakan yang dimiliki Jorge sekarang. Nama apa pun yang dimulai dengan `flow-logs-` diperbolehkan. Untuk contoh ini, masukkan **`flow-logs-for-jorge`** nama Peran.
14. (Opsional) Untuk Deskripsi peran, masukkan deskripsi.
15. Setelah meninjau perinciannya, pilih Buat peran.
16. Sekarang, Anda dapat mengonfigurasi kebijakan kepercayaan yang diperlukan untuk skenario ini. Pada halaman Peran, pilih flow-logs-for-jorgeperan (nama, bukan kotak centang). Pada halaman perincian peran baru Anda, pilih Hubungan kepercayaan, lalu pilihEdit hubungan kepercayaan.
17. Ubah baris "Layanan" agar dibaca sebagai berikut, menggantikan entri untuk `ec2.amazonaws.com`:

```
        "Service": "vpc-flow-logs.amazonaws.com"
```

18. Jorge sekarang dapat membuat log aliran untuk subnet VPC atau subnet di konsol AmazonEC2. Saat Anda membuat log alur, tentukan flow-logs-for-jorgeperannya. Peran tersebut memiliki izin untuk membuat data log dan menuliskan data ke log itu.

AWS kunci konteks kondisi global

Ketika kepala sekolah membuat permintaan AWS, AWS mengumpulkan informasi permintaan ke dalam konteks permintaan. Anda dapat menggunakan `Condition` elemen JSON kebijakan untuk membandingkan kunci dalam konteks permintaan dengan nilai kunci yang Anda tentukan dalam kebijakan Anda. Informasi permintaan disediakan oleh sumber yang berbeda, termasuk prinsipal yang membuat permintaan, sumber daya permintaan dibuat terhadap, dan metadata tentang permintaan itu sendiri.

Kunci kondisi global dapat digunakan di semua AWS layanan. Meskipun kunci kondisi ini dapat digunakan di semua kebijakan, kunci tidak tersedia di setiap konteks permintaan. Misalnya, kunci `aws:SourceAccount` kondisi hanya tersedia ketika panggilan ke sumber daya Anda dilakukan langsung oleh kepala AWS layanan. Untuk mempelajari lebih lanjut tentang keadaan di mana kunci global disertakan dalam konteks permintaan, lihat Informasi ketersediaan untuk setiap kunci.

Beberapa layanan individu membuat kunci kondisi mereka sendiri yang tersedia dalam konteks permintaan untuk layanan lain. Kunci kondisi lintas layanan adalah jenis kunci kondisi global yang menyertakan awalan yang cocok dengan nama layanan, seperti `ec2:` atau`lambda:`, tetapi tersedia di seluruh layanan lain.

Kunci kondisi khusus layanan didefinisikan untuk digunakan dengan layanan individual AWS . Misalnya, Amazon S3 memungkinkan Anda menulis kebijakan dengan kunci `s3:VersionId` kondisi untuk membatasi akses ke versi tertentu dari objek Amazon S3. Kunci kondisi ini unik untuk layanan, artinya hanya berfungsi dengan permintaan ke layanan Amazon S3. Untuk kunci kondisi yang spesifik layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan yang kuncinya ingin Anda lihat.

> **Note**
>
> Jika Anda menggunakan tombol kondisi yang hanya tersedia dalam beberapa keadaan, Anda dapat menggunakan IfExistsversi operator kondisi. Jika kunci kondisi hilang dari konteks permintaan, kebijakan dapat gagal dalam evaluasi. Misalnya, gunakan blok kondisi berikut dengan `...IfExists` operator untuk mencocokkan ketika permintaan berasal dari rentang IP tertentu atau dari yang spesifikVPC. Jika salah satu atau kedua kunci tidak

disertakan dalam konteks permintaan, kondisi masih akan mengembalikan `true`. Nilai hanya diperiksa jika kunci yang ditentukan disertakan dalam konteks permintaan. Untuk informasi selengkapnya tentang cara kebijakan dievaluasi ketika kunci tidak ada untuk operator lain, lihat Operator kondisi.

```
"Condition": {
    "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
    "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
}
```

Important

Untuk membandingkan kondisi Anda terhadap konteks permintaan dengan beberapa nilai kunci, Anda harus menggunakan operator kumpulan `ForAllValues` atau `ForAnyValue`. Gunakan operator set hanya dengan kunci kondisi multivalued. Jangan gunakan operator set dengan kunci syarat bernilai tunggal. Untuk informasi selengkapnya, lihat Kunci konteks multivaluasi.

Sifat-sifat kepala sekolah	Properti sesi peran	Properti jaringan	Properti dari sumber daya	Properti permintaan
aws: PrincipalArn	aws: FederatedProvider	aws: SourceIp	aws: ResourceAccount	aws: CalledVia
aws: PrincipalAccount	aws: TokenIssueTime	aws: SourceVpc	aws: ResourceOrg ID	aws: CalledViaFirst
aws: PrincipalOrgPaths	aws: MultiFactorAuthAge	aws: SourceVpce	aws: ResourceOrgPaths	aws: CalledViaLast
aws: PrincipalOrg ID	aws: MultiFactorAuthPresent	aws: VpcSourceIp	aws:ResourceTag/tag-kunci	AWS: ViaAWSService
aws:PrincipalTag/tag-kunci	AWS: EC2InstanceSourceVpc			aws: CurrentTime
				aws: EpochTime

Sifat-sifat kepala sekolah	Properti sesi peran	Properti jaringan	Properti dari sumber daya	Properti permintaan
aws: PrincipalIsAWSService aws: PrincipalServiceName aws: PrincipalServiceNamesList aws: PrincipalType aws:userid aws:username	AWS: EC2InstanceSourcePrivateIPv4 aws: SourceIdentity EC2: RoleDelivery EC2: SourceInstanceArn lem: RoleAssumedBy lem: CredentialIssuingService lambda: SourceFunctionArn ssm: SourceInstanceArn toko identitas: UserId			aws:referer aws: RequestedRegion aws:RequestTag/tag-kunci aws: TagKeys aws: SecureTransport aws: SourceArn aws: SourceAccount aws: SourceOrgPaths aws: SourceOrgID aws: UserAgent

Kunci kondisi sensitif

Kunci kondisi berikut dianggap sensitif karena nilainya dihasilkan oleh mesin. Penggunaan wildcard dalam kunci kondisi ini tidak memiliki kasus penggunaan yang valid, bahkan dengan substring dari nilai kunci dengan wildcard. Ini karena wildcard dapat mencocokkan kunci kondisi dengan nilai apa pun, yang dapat menimbulkan risiko keamanan.

- aws: PrincipalAccount
- aws: PrincipalOrg ID
- aws: ResourceAccount
- aws: ResourceOrg ID
- aws: SourceAccount
- aws: SourceOrg ID
- aws: SourceVpc
- aws: SourceVpce
- aws:userid

Sifat-sifat kepala sekolah

Gunakan tombol kondisi berikut untuk membandingkan detail tentang prinsipal yang membuat permintaan dengan properti utama yang Anda tentukan dalam kebijakan. Untuk daftar kepala sekolah yang dapat membuat permintaan, lihat. Cara menentukan kepala sekolah

Daftar Isi

- aws: PrincipalArn
- aws: PrincipalAccount
- aws: PrincipalOrgPaths
- aws: PrincipalOrg ID
- aws:PrincipalTag/tag-kunci
- aws: PrincipalIs AWSService
- aws: PrincipalServiceName
- aws: PrincipalServiceNamesList
- aws: PrincipalType
- aws:userid
- aws:username

aws: PrincipalArn

Gunakan kunci ini untuk membandingkan Amazon Resource Name (ARN) prinsipal yang membuat permintaan dengan ARN yang Anda tentukan dalam kebijakan. Untuk IAM peran, konteks permintaan mengembalikan peran, bukan ARN dari pengguna yang mengambil peran. ARN

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan untuk semua permintaan yang ditandatangani. Permintaan anonim tidak menyertakan kunci ini. Anda dapat menentukan jenis prinsipal berikut dalam kunci kondisi ini:
  - IAMperan
  - IAMpengguna
  - AWS STS sesi pengguna federasi
  - Akun AWS pengguna root
- Tipe data -ARN, String

  AWS merekomendasikan agar Anda menggunakan ARNoperator alih-alih operator string saat membandingkanARNs.
- Jenis nilai - Bernilai tunggal
- Contoh nilai Daftar berikut menunjukkan nilai konteks permintaan yang dikembalikan untuk berbagai jenis prinsipal yang dapat Anda tentukan dalam kunci kondisi: `aws:PrincipalArn`
  - IAMperan - Konteks permintaan berisi nilai berikut untuk kunci kondisi`aws:PrincipalArn`. Jangan tentukan sesi peran yang diasumsikan ARN sebagai nilai untuk kunci kondisi ini. Untuk informasi lebih lanjut tentang kepala sesi peran yang diasumsikan, lihatKepala sesi peran.

    ```
    arn:aws:iam::123456789012:role/role-name
    ```
  - IAMuser - Konteks permintaan berisi nilai berikut untuk kunci kondisi`aws:PrincipalArn`.

    ```
    arn:aws:iam::123456789012:user/user-name
    ```
  - AWS STS sesi pengguna federasi - Konteks permintaan berisi nilai berikut untuk kunci `aws:PrincipalArn` kondisi.

    ```
    arn:aws:sts::123456789012:federated-user/user-name
    ```
  - Akun AWS root user - Konteks permintaan berisi nilai berikut untuk kunci kondisi`aws:PrincipalArn`. Ketika Anda menentukan pengguna root ARN sebagai nilai untuk

kunci `aws:PrincipalArn` kondisi, itu membatasi izin hanya untuk pengguna root dari Akun AWS. Ini berbeda dengan menentukan pengguna root ARN dalam elemen utama kebijakan berbasis sumber daya, yang mendelegasikan wewenang ke. Akun AWS Untuk informasi selengkapnya tentang menentukan pengguna root ARN dalam elemen utama kebijakan berbasis sumber daya, lihat. Akun AWS utama

```
arn:aws:iam::123456789012:root
```

Anda dapat menentukan pengguna root ARN sebagai nilai untuk kunci kondisi `aws:PrincipalArn` dalam kebijakan kontrol AWS Organizations layanan (SCPs). SCPsadalah jenis kebijakan organisasi yang digunakan untuk mengelola izin di organisasi Anda dan hanya memengaruhi akun anggota di organisasi. SCPMembatasi izin untuk IAM pengguna dan peran dalam akun anggota, termasuk pengguna root akun anggota. Untuk informasi selengkapnya tentang efek SCPs pada izin, lihat SCPefek pada izin di Panduan Pengguna Organizations.

aws: PrincipalAccount

Gunakan kunci ini untuk membandingkan akun yang memiliki prinsipal yang meminta dengan pengenal akun yang Anda tentukan dalam kebijakan. Untuk permintaan anonim, konteks permintaan akan ditampilkan`anonymous`.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Dalam contoh berikut, akses ditolak kecuali ke kepala sekolah dengan nomor rekening. `123456789012`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:accountID:resource"
```

```
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws: PrincipalOrgPaths

Gunakan kunci ini untuk membandingkan AWS Organizations jalur bagi kepala sekolah yang membuat permintaan ke jalur dalam kebijakan. Prinsipal itu dapat berupa IAM pengguna, IAM peran, pengguna federasi, atau Pengguna root akun AWS. Dalam kebijakan, kunci kondisi ini memastikan bahwa pemohon adalah anggota akun dalam akar organisasi tertentu atau unit organisasi (OUs) di AWS Organizations. AWS Organizations Path adalah representasi teks dari struktur entitas Organizations. Untuk informasi lebih lanjut tentang cara menggunakan dan memahami jalur, lihat Memahami jalur entitas AWS Organizations.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal merupakan anggota dari suatu organisasi. Permintaan anonim tidak menyertakan kunci ini.
- Tipe data - String (daftar)
- Jenis nilai — Multivalued

> **Note**
>
> Organisasi IDs secara global unik tetapi OU IDs dan root IDs hanya unik dalam suatu organisasi. Ini berarti bahwa tidak ada dua organisasi yang memiliki ID organisasi yang sama. Namun, organisasi lain mungkin memiliki OU atau root dengan ID yang sama seperti milik Anda. Kami merekomendasikan agar Anda selalu menyertakan ID organisasi saat menentukan OU atau root.

Misalnya, kondisi berikut mengembalikan `true` prinsipal di akun yang dilampirkan langsung ke `ou-ab12-22222222` OU, tetapi tidak pada anaknya. OUs

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

Kondisi berikut kembali `true` untuk prinsipal dalam akun yang dilampirkan langsung ke OU atau salah satu anaknya. OUs Ketika Anda menyertakan wildcard, Anda harus menggunakan operator kondisi `StringLike`.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

Kondisi berikut kembali `true` untuk prinsipal dalam akun yang dilampirkan langsung ke salah satu anakOUs, tetapi tidak langsung ke OU orang tua. Kondisi sebelumnya ditujukan untuk OU atau turunannya. Kondisi berikut hanya ditujukan untuk turunannya (dan turunan dari turunan tersebut).

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

Kondisi berikut memungkinkan akses untuk setiap prinsipal dalam organisasi `o-a1b2c3d4e5`, terlepas dari OU indukannya.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` adalah kunci kondisi multinilai. Kunci multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Saat Anda menggunakan beberapa nilai dengan operator kondisi `ForAnyValue`, jalur prinsipal harus sesuai dengan salah satu jalur yang tercantum dalam kebijakan. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Kunci konteks multivaluasi.

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
```

```
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

aws: PrincipalOrg ID

Gunakan kunci ini untuk membandingkan pengenal organisasi tempat prinsipal permintaan berada dengan pengenal yang ditentukan dalam kebijakan. AWS Organizations

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal merupakan anggota dari suatu organisasi. Permintaan anonim tidak menyertakan kunci ini.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Kunci global ini memberikan alternatif untuk mencantumkan semua akun IDs untuk semua AWS akun dalam suatu organisasi. Anda dapat menggunakan tombol kondisi ini untuk menyederhanakan penentuan elemen `Principal` dalam kebijakan berbasis sumber daya. Anda dapat menentukan ID organisasi dalam elemen kondisi. Saat Anda menambah dan menghapus akun, kebijakan yang mencakup kunci `aws:PrincipalOrgID` akan secara otomatis menyertakan akun yang benar dan tidak memerlukan pembaruan manual.

Misalnya, kebijakan bucket Amazon S3 berikut memungkinkan anggota akun mana pun di organisasi `o-xxxxxxxxxxx` untuk menambahkan objek ke dalam bucket `amzn-s3-demo-bucket`.

```
{
 "Version": "2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
   "Condition": {"StringEquals":
     {"aws:PrincipalOrgID":"o-xxxxxxxxxxx"}
   }
 }
}
```

> **Note**
>
> Kondisi global ini juga berlaku untuk akun manajemen suatu AWS organisasi. Kebijakan ini mencegah semua kepala sekolah di luar organisasi tertentu mengakses bucket Amazon S3. Ini termasuk AWS layanan apa pun yang berinteraksi dengan sumber daya internal Anda, seperti AWS CloudTrail mengirim data log ke bucket Amazon S3 Anda. Untuk mempelajari bagaimana Anda dapat memberikan akses AWS layanan dengan aman, lihataws: PrincipalIs AWSService.

Untuk informasi lebih lanjut tentang AWS Organizations, lihat Apa itu AWS Organizations? dalam AWS Organizations User Guide.

aws:PrincipalTag/tag-kunci

Gunakan kunci ini untuk membandingkan tanda yang dilampirkan ke prinsipal yang mengajukan permintaan dengan tanda yang Anda tentukan dalam kebijakan. Jika prinsipal memiliki lebih dari satu tanda yang dilampirkan, konteks permintaan mencakup satu kunci `aws:PrincipalTag` untuk setiap kunci tanda yang dilampirkan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan jika prinsipal menggunakan IAM pengguna dengan tag terlampir. Ini termasuk untuk prinsipal menggunakan IAM peran dengan tag terlampir atau tag sesi. Permintaan anonim tidak menyertakan kunci ini.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Anda dapat menambahkan atribut khusus ke pengguna atau peran dalam bentuk pasangan nilai kunci. Untuk informasi selengkapnya tentang IAM tag, lihatTag untuk AWS Identity and Access Management sumber daya. Anda dapat menggunakan `aws:PrincipalTag` untuk mengontrol akses untuk prinsipal AWS .

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna dengan **`department=hr`** tag mengelola IAM pengguna, grup, atau peran. Untuk menggunakan kebijakan ini, ganti *`italicized placeholder text`* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

aws: PrincipalIs AWSService

Gunakan kunci ini untuk memeriksa apakah panggilan ke sumber daya Anda dilakukan langsung oleh kepala AWS layanan. Misalnya, AWS CloudTrail menggunakan prinsip layanan `cloudtrail.amazonaws.com` untuk menulis log ke bucket Amazon S3 Anda. Kunci konteks permintaan diatur ke benar ketika layanan menggunakan prinsipal layanan untuk melakukan tindakan langsung pada sumber daya Anda. Kunci konteks disetel ke false jika layanan menggunakan kredensi IAM prinsipal untuk membuat permintaan atas nama kepala sekolah. Ini juga disetel ke false jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah.

- Ketersediaan — Kunci ini hadir dalam konteks permintaan untuk semua API permintaan yang ditandatangani yang menggunakan AWS kredensil. Permintaan anonim tidak menyertakan kunci ini.
- Tipe data — Boolean
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk membatasi akses ke identitas tepercaya dan lokasi jaringan yang diharapkan sambil memberikan akses ke AWS layanan dengan aman.

Dalam contoh kebijakan bucket Amazon S3 berikut, akses ke bucket dibatasi kecuali permintaan tersebut berasal dari `vpc-111bbb22` atau berasal dari prinsipal layanan, seperti. CloudTrail

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

Dalam video berikut, pelajari selengkapnya tentang cara menggunakan kunci kondisi `aws:PrincipalIsAWSService` dalam kebijakan.

Berikan akses secara aman ke pengguna resmi Anda, lokasi jaringan yang diharapkan, dan AWS layanan secara bersamaan.

aws: PrincipalServiceName

Gunakan kunci ini untuk membandingkan nama prinsipal layanan dalam kebijakan dengan prinsipal layanan yang membuat permintaan untuk sumber daya Anda. Anda dapat menggunakan kunci ini untuk memeriksa apakah panggilan ini dibuat oleh prinsipal layanan tertentu. Ketika prinsipal layanan membuat permintaan langsung ke sumber daya Anda,kunci `aws:PrincipalServiceName`berisi nama prinsipal layanan. Misalnya, nama utama AWS CloudTrail layanan adalah`cloudtrail.amazonaws.com`.

- Ketersediaan — Kunci ini ada dalam permintaan saat panggilan dilakukan oleh kepala AWS layanan. Kunci ini tidak ada dalam situasi lain, termasuk berikut ini:
  - Jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah.
  - Jika layanan menggunakan kredensi IAM kepala sekolah untuk mengajukan permintaan atas nama kepala sekolah.

  - Jika panggilan dilakukan langsung oleh IAM kepala sekolah.
  - Jika panggilan dilakukan oleh pemohon anonim.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk membatasi akses ke identitas tepercaya dan lokasi jaringan yang diharapkan sambil memberikan akses ke layanan dengan aman. AWS

Dalam contoh kebijakan bucket Amazon S3 berikut, akses ke bucket dibatasi kecuali permintaan tersebut berasal dari `vpc-111bbb22` atau berasal dari prinsipal layanan, seperti. CloudTrail

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
        }
      }
    }
  ]
}
```

aws: PrincipalServiceNamesList

Kunci ini menyediakan daftar semua nama prinsipal layanan yang termasuk dalam layanan. Ini adalah kunci kondisi lanjutan. Anda dapat menggunakannya untuk membatasi layanan dari mengakses sumber daya Anda hanya dari Wilayah tertentu. Beberapa layanan dapat membuat prinsipal layanan Regional untuk menunjukkan instans layanan tertentu dalam Wilayah tertentu. Anda dapat membatasi akses ke sumber daya untuk instans layanan tertentu. Saat prinsipal layanan membuat permintaan langsung ke sumber daya Anda, `aws:PrincipalServiceNamesList` berisi

daftar seluruh nama prinsipal layanan yang tidak diurutkan yang terkait dengan instans Regional layanan.

- Ketersediaan — Kunci ini ada dalam permintaan saat panggilan dilakukan oleh kepala AWS layanan. Kunci ini tidak ada dalam situasi lain, termasuk berikut ini:
  - Jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah.
  - Jika layanan menggunakan kredensi IAM kepala sekolah untuk mengajukan permintaan atas nama kepala sekolah.
  - Jika panggilan dilakukan langsung oleh IAM kepala sekolah.
  - Jika panggilan dilakukan oleh pemohon anonim.
- Tipe data - String (daftar)
- Jenis nilai — Multivalued

`aws:PrincipalServiceNamesList` adalah kunci kondisi multinilai. Kunci multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan `ForAnyValue` atau `ForAllValues` mengatur operator dengan operator kondisi string untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Kunci konteks multivaluasi.

aws: PrincipalType

Gunakan kunci ini untuk membandingkan tipe prinsipal yang membuat permintaan dengan tipe prinsipal yang Anda sebutkan dalam kebijakan. Untuk informasi selengkapnya, lihat Cara menentukan kepala sekolah. Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihatNilai-nilai kunci utama.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

aws:userid

Gunakan kunci ini untuk membandingkan pengenal prinsipal pemohon dengan ID yang Anda tentukan dalam kebijakan. Untuk IAM pengguna, nilai konteks permintaan adalah ID pengguna. Untuk IAM peran, format nilai ini dapat bervariasi. Untuk detail tentang bagaimana informasi tersebut

muncul untuk prinsipal yang berbeda, lihat Cara menentukan kepala sekolah. Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihatNilai-nilai kunci utama.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

aws:username

Gunakan kunci ini untuk membandingkan nama pengguna pemohon dengan nama pengguna yang Anda sebutkan dalam kebijakan. Untuk detail tentang bagaimana informasi tersebut muncul untuk prinsipal yang berbeda, lihat Cara menentukan kepala sekolah. Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihatNilai-nilai kunci utama.

- Ketersediaan — Kunci ini selalu disertakan dalam konteks permintaan untuk IAM pengguna. Permintaan dan permintaan anonim yang dibuat menggunakan IAM peran Pengguna root akun AWS atau tidak menyertakan kunci ini. Permintaan yang dibuat menggunakan kredensil Pusat IAM Identitas tidak menyertakan kunci ini dalam konteksnya.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Properti sesi peran

Gunakan tombol kondisi berikut untuk membandingkan properti sesi peran pada saat sesi dibuat. Kunci kondisi ini hanya tersedia jika permintaan dibuat oleh prinsipal dengan sesi peran atau kredenal pengguna gabungan. Nilai untuk kunci kondisi ini disematkan dalam token sesi peran.

Peran adalah jenis kepala sekolah. Anda juga dapat menggunakan kunci kondisi dari Sifat-sifat kepala sekolah bagian untuk mengevaluasi properti peran saat peran membuat permintaan.

Daftar Isi

- aws: ChatbotSourceArn
- AWS: EC2 InstanceSourceVpc
- AWS: EC2 InstanceSourcePrivate IPv4
- aws: SourceIdentity
- EC2: RoleDelivery
- EC2: SourceInstanceArn
- lem: RoleAssumedBy
- lem: CredentialIssuingService
- lambda: SourceFunctionArn
- ssm: SourceInstanceArn
- toko identitas: UserId

aws: FederatedProvider

Gunakan kunci ini untuk membandingkan penyedia identitas penerbit (iDP) prinsipal dengan iDP yang Anda tentukan dalam kebijakan. Ini berarti bahwa IAM peran diasumsikan menggunakan `AssumeRoleWithWebIdentity` AWS STS operasi. Bila kredensi sementara sesi peran yang dihasilkan digunakan untuk membuat permintaan, konteks permintaan mengidentifikasi IDP yang mengautentikasi identitas federasi asli.

- Ketersediaan — Kunci ini hadir ketika prinsipal adalah kepala sesi peran dan sesi itu dikeluarkan ketika peran diasumsikan`AssumeRoleWithWebIdentity`.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Misalnya, jika pengguna diautentikasi melalui Amazon Cognito, konteks permintaan menyertakan nilainya. `cognito-identity.amazonaws.com` Demikian pula, jika pengguna diautentikasi melalui Login with Amazon, konteks permintaan menyertakan nilainya`www.amazon.com`.

Anda dapat menggunakan kunci kondisi bernilai tunggal apa pun sebagai variabel. Contoh kebijakan berbasis sumber daya berikut menggunakan `aws:FederatedProvider` kunci sebagai variabel kebijakan dalam sumber daya. ARN Kebijakan ini memungkinkan setiap prinsipal yang melakukan autentikasi menggunakan iDP untuk mengeluarkan objek dari bucket Amazon S3 dengan jalur yang khusus untuk penyedia identitas penerbit.

aws: TokenIssueTime

Gunakan kunci ini untuk membandingkan tanggal dan waktu saat kredensial keamanan sementara dikeluarkan dengan tanggal dan waktu yang Anda tentukan dalam kebijakan.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal menggunakan kredensial sementara untuk membuat permintaan. Kuncinya tidak ada di AWS CLI, AWS API, atau AWS SDK permintaan yang dibuat menggunakan kunci akses.
- Tipe data - Tanggal
- Jenis nilai - Bernilai tunggal

Untuk mempelajari layanan mana yang mendukung penggunaan kredensial sementara, lihat AWS layanan yang bekerja dengan IAM.

aws: MultiFactorAuthAge

Gunakan kunci ini untuk membandingkan jumlah detik sejak prinsipal permintaan diotorisasi menggunakan nomor MFA yang Anda tentukan dalam kebijakan. Untuk informasi lebih lanjut tentangMFA, lihatAWS Otentikasi multi-faktor di IAM.

> **Important**
>
> Kunci kondisi ini tidak ada untuk identitas gabungan atau permintaan yang dibuat menggunakan kunci akses untuk menandatangani AWS CLI, AWS API, atau AWS SDK permintaan. Untuk mempelajari selengkapnya tentang menambahkan MFA perlindungan ke API operasi dengan kredensil keamanan sementara, lihat. APIAkses aman dengan MFA Untuk memeriksa MFA apakah digunakan untuk memvalidasi identitas IAM federasi, Anda dapat meneruskan metode otentikasi dari penyedia identitas Anda ke tag sesi AWS . Untuk detailnya, lihat Lulus tag sesi di AWS STS. MFAUntuk menerapkan IAM identitas Pusat Identitas, Anda dapat mengaktifkan atribut untuk kontrol akses untuk meneruskan klaim SAML pernyataan dengan metode otentikasi dari penyedia identitas Anda ke Pusat Identitas. IAM

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal menggunakan kredensil keamanan sementara untuk membuat permintaan. Kebijakan dengan MFA ketentuan dapat dilampirkan ke:
  - Pengguna atau grup IAM

  - Sumber daya seperti bucket Amazon S3, SQS antrian Amazon, atau topik Amazon SNS
  - Kebijakan kepercayaan dari IAM peran yang dapat diasumsikan oleh pengguna
- Tipe data - Numerik
- Jenis nilai - Bernilai tunggal

aws: MultiFactorAuthPresent

Gunakan kunci ini untuk memeriksa apakah otentikasi multi-faktor (MFA) digunakan untuk memvalidasi kredenal keamanan sementara yang membuat permintaan.

> **Important**
>
> Kunci kondisi ini tidak ada untuk identitas gabungan atau permintaan yang dibuat menggunakan kunci akses untuk menandatangani AWS CLI, AWS API, atau AWS SDK permintaan. Untuk mempelajari selengkapnya tentang menambahkan MFA perlindungan ke API operasi dengan kredensil keamanan sementara, lihat. APIAkses aman dengan MFA Untuk memeriksa MFA apakah digunakan untuk memvalidasi identitas IAM federasi, Anda dapat meneruskan metode otentikasi dari penyedia identitas Anda ke tag sesi AWS . Untuk detailnya, lihat Lulus tag sesi di AWS STS. MFAUntuk menerapkan IAM identitas Pusat Identitas, Anda dapat mengaktifkan atribut untuk kontrol akses untuk meneruskan klaim SAML pernyataan dengan metode otentikasi dari penyedia identitas Anda ke Pusat Identitas. IAM

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal menggunakan kredensial sementara untuk membuat permintaan. Kebijakan dengan MFA ketentuan dapat dilampirkan ke:
  - Pengguna atau grup IAM
  - Sumber daya seperti bucket Amazon S3, SQS antrian Amazon, atau topik Amazon SNS
  - Kebijakan kepercayaan dari IAM peran yang dapat diasumsikan oleh pengguna
- Tipe data — Boolean
- Jenis nilai - Bernilai tunggal

Kredensi sementara digunakan untuk mengautentikasi IAM peran dan IAM pengguna dengan token sementara dari AssumeRoleatau GetSessionToken, dan pengguna. AWS Management Console

IAMkunci akses pengguna adalah kredensil jangka panjang, tetapi dalam beberapa kasus, AWS membuat kredensil sementara atas nama IAM pengguna untuk melakukan operasi. Dalam kasus ini, kunci `aws:MultiFactorAuthPresent` tersedia dalam permintaan dan diatur ke nilai `false`. Ada dua kasus umum hal ini bisa terjadi:

- IAMpengguna di AWS Management Console tanpa sadar menggunakan kredensi sementara. Pengguna masuk ke konsol menggunakan nama pengguna dan kata sandi yang merupakan kredensial jangka panjang. Namun, di latar belakang, konsol membuat kredensial sementara atas nama pengguna.
- Jika IAM pengguna melakukan panggilan ke AWS layanan, layanan menggunakan kembali kredensi pengguna untuk membuat permintaan lain ke layanan yang berbeda. Misalnya, saat memanggil Athena untuk mengakses bucket Amazon S3, atau saat AWS CloudFormation menggunakan untuk membuat instance Amazon. EC2 Untuk permintaan berikutnya, AWS gunakan kredensyal sementara.

Untuk mempelajari layanan mana yang mendukung penggunaan kredensial sementara, lihat AWS layanan yang bekerja dengan IAM.

`aws:MultiFactorAuthPresent`Kuncinya tidak ada ketika CLI perintah API atau dipanggil dengan kredensil jangka panjang, seperti pasangan kunci akses pengguna. Oleh karena itu, kami merekomendasikan bahwa ketika Anda memeriksa kunci ini, Anda menggunakan versi `...IfExists` dari operator kondisi.

Penting untuk dipahami bahwa `Condition` elemen berikut bukanlah cara yang dapat diandalkan untuk memeriksa apakah permintaan diautentikasi menggunakanMFA.

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Kombinasi `Deny` efek, `Bool` elemen, dan `false` nilai ini menyangkal permintaan yang dapat diautentikasi menggunakanMFA, tetapi tidak. Ini hanya berlaku untuk kredenal sementara yang mendukung penggunaan. MFA Pernyataan ini tidak menolak akses ke permintaan yang dibuat menggunakan kredensi jangka panjang, atau permintaan yang diautentikasi menggunakan. MFA Gunakan contoh ini dengan hati-hati karena logikanya rumit dan tidak menguji apakah MFA - authentication benar-benar digunakan.

Jangan juga menggunakan kombinasi efek `Deny`, elemen `Null`, dan `true` karena contoh berperilaku dengan cara yang sama dan logikanya bahkan lebih rumit.

Kombinasi yang Direkomendasikan

Kami menyarankan Anda menggunakan `BoolIfExists`operator untuk memeriksa apakah permintaan diautentikasi menggunakanMFA.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Kombinasi ini`Deny`,`BoolIfExists`, dan `false` menolak permintaan yang tidak diautentikasi menggunakan. MFA Secara khusus, ia menolak permintaan dari kredensil sementara yang tidak termasuk. MFA Ini juga menyangkal permintaan yang dibuat menggunakan kredensil jangka panjang, seperti AWS CLI atau AWS API operasi yang dibuat menggunakan kunci akses. Operator `*IfExists` memeriksa keberadaan kunci `aws:MultiFactorAuthPresent` dan apakah bisa tersedia atau tidak, sebagaimana yang ditunjukkan oleh keberadaannya. Gunakan ini ketika Anda ingin menolak permintaan apa pun yang tidak diautentikasi menggunakanMFA. Ini lebih aman, tetapi dapat merusak kode atau skrip apa pun yang menggunakan kunci akses untuk mengakses AWS CLI atau AWS API.

Kombinasi Alternatif

Anda juga dapat menggunakan `BoolIfExists`operator untuk mengizinkan permintaan yang MFA diautentikasi dan AWS CLI atau AWS API permintaan yang dibuat menggunakan kredensi jangka panjang.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Kondisi ini cocok jika kunci ada dan tersedia atau jika kunci tidak ada. Kombinasi`Allow`,`BoolIfExists`, dan `true` memungkinkan permintaan yang diautentikasi menggunakanMFA, atau permintaan yang tidak dapat diautentikasi menggunakan. MFA Ini berarti bahwa AWS CLI, AWS API, dan AWS SDK operasi diperbolehkan ketika pemohon menggunakan kunci akses jangka panjang mereka. Kombinasi ini tidak mengizinkan permintaan dari kredensi sementara yang bisa, tetapi tidak termasuk. MFA

Saat Anda membuat kebijakan menggunakan editor visual IAM konsol dan memilih MFAyang diperlukan, kombinasi ini diterapkan. Pengaturan ini memerlukan MFA akses konsol, tetapi memungkinkan akses terprogram tanpaMFA.

Atau, Anda dapat menggunakan `Bool` operator untuk mengizinkan permintaan terprogram dan konsol hanya ketika diautentikasi menggunakan. MFA

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Kombinasi dari`Allow`,`Bool`, dan hanya `true` memungkinkan permintaan yang MFA diautentikasi. Ini hanya berlaku untuk kredenal sementara yang mendukung penggunaan. MFA Pernyataan ini tidak mengizinkan akses ke permintaan yang dibuat menggunakan kunci akses jangka panjang, atau permintaan yang dibuat menggunakan kredensial sementara tanpa. MFA

Jangan gunakan konstruksi kebijakan yang mirip dengan berikut ini untuk memeriksa apakah MFA kunci tersebut ada:

```
#####   WARNING: USE WITH CAUTION   #####

"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Kombinasi `Allow` efek, `Null` elemen, dan `false` nilai ini hanya memungkinkan permintaan yang dapat diautentikasi menggunakanMFA, terlepas dari apakah permintaan tersebut benar-benar diautentikasi. Ini memungkinkan semua permintaan yang dibuat menggunakan kredensial sementara, dan menolak akses untuk kredensial jangka panjang. Gunakan contoh ini dengan hati-hati karena tidak menguji apakah MFA -authentication benar-benar digunakan.

aws: ChatbotSourceArn

Gunakan kunci ini untuk membandingkan konfigurasi obrolan sumber yang ARN ditetapkan oleh prinsipal dengan konfigurasi obrolan yang ARN Anda tentukan dalam kebijakan IAM peran yang terkait dengan konfigurasi saluran Anda. Anda dapat mengotorisasi permintaan berdasarkan sesi peran asumsikan yang dimulai oleh. AWS Chatbot

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan oleh AWS Chatbot layanan setiap kali sesi peran diasumsikan. Nilai kuncinya adalah konfigurasi obrolanARN, seperti saat Anda menjalankan AWS CLI perintah dari saluran obrolan.
- Tipe data - ARN
- Jenis nilai - Bernilai tunggal
- Nilai contoh - `arn:aws::chatbot::123456789021:chat-configuration/slack-channel/private_channel`

Kebijakan berikut menolak permintaan Amazon S3 menempatkan pada bucket yang ditentukan untuk semua permintaan yang berasal dari saluran Slack.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleS3Deny",
            "Effect": "Deny",
            "Action": "s3:PutObject",
            "Resource": "arn:aws::s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "StringLike": {
                      "aws:ChatbotSourceArn": "arn:aws::chatbot::*:chat-configuration/
slack-channel/*"
                }
            }
        }
    ]
}
```

AWS: EC2 InstanceSourceVpc

Kunci ini mengidentifikasi VPC ke mana kredensi EC2 IAM peran Amazon dikirimkan. Anda dapat menggunakan kunci ini dalam kebijakan dengan kunci `aws:SourceVPC`global untuk memeriksa apakah panggilan dibuat dari VPC (`aws:SourceVPC`) yang cocok dengan VPC tempat kredensi dikirimkan ke (`aws:Ec2InstanceSourceVpc`).

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali pemohon menandatangani permintaan dengan kredensi EC2 peran Amazon. Ini dapat digunakan dalam IAM kebijakan, kebijakan kontrol layanan, kebijakan VPC titik akhir, dan kebijakan sumber daya.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Kunci ini dapat digunakan dengan nilai VPC pengenal, tetapi paling berguna bila digunakan sebagai variabel yang dikombinasikan dengan kunci `aws:SourceVpc` konteks. Kunci `aws:SourceVpc` konteks disertakan dalam konteks permintaan hanya jika pemohon menggunakan VPC titik akhir untuk membuat permintaan. Menggunakan `aws:Ec2InstanceSourceVpc` dengan

`aws:SourceVpc` memungkinkan Anda untuk menggunakan `aws:Ec2InstanceSourceVpc` lebih luas karena membandingkan nilai yang biasanya berubah bersama.

> **Note**
>
> Kunci kondisi ini tidak tersedia di EC2 -Classic.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Pada contoh di atas, akses ditolak jika `aws:SourceVpc` nilainya tidak sama dengan `aws:Ec2InstanceSourceVpc` nilainya. Pernyataan kebijakan dibatasi hanya pada peran yang digunakan sebagai peran EC2 instans Amazon dengan menguji keberadaan kunci `ec2:SourceInstanceARN` kondisi.

Kebijakan ini digunakan `aws:ViaAWSService` AWS untuk mengizinkan otorisasi permintaan saat permintaan dibuat atas nama peran EC2 instans Amazon Anda. Misalnya, saat Anda membuat permintaan dari EC2 instans Amazon ke bucket Amazon S3 terenkripsi, Amazon S3 melakukan panggilan atas nama Anda. AWS KMS Beberapa kunci tidak ada saat permintaan dibuat AWS KMS.

AWS: EC2 InstanceSourcePrivate IPv4

Kunci ini mengidentifikasi IPv4 alamat pribadi dari elastisitas network interface utama tempat kredensi EC2 IAM peran Amazon dikirimkan. Anda harus menggunakan kunci kondisi ini dengan kunci pendampingnya `aws:Ec2InstanceSourceVpc` untuk memastikan bahwa Anda memiliki kombinasi VPC ID dan IP pribadi sumber yang unik secara global. Gunakan kunci ini `aws:Ec2InstanceSourceVpc` untuk memastikan bahwa permintaan dibuat dari alamat IP pribadi yang sama dengan yang dikirimkan kredensialnya.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali pemohon menandatangani permintaan dengan kredensi EC2 peran Amazon. Ini dapat digunakan dalam IAM kebijakan, kebijakan kontrol layanan, kebijakan VPC titik akhir, dan kebijakan sumber daya.
- Tipe data — alamat IP
- Jenis nilai - Bernilai tunggal

> **Important**
>
> Kunci ini tidak boleh digunakan sendiri dalam sebuah `Allow` pernyataan. Alamat IP pribadi menurut definisi tidak unik secara global. Anda harus menggunakan `aws:Ec2InstanceSourceVpc` kunci setiap kali Anda menggunakan `aws:Ec2InstanceSourcePrivateIPv4` kunci untuk menentukan kredensi EC2 instans Amazon VPC Anda dapat digunakan.

> **Note**
>
> Kunci kondisi ini tidak tersedia di EC2 -Classic.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
```

```
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        }
    ]
}
```

aws: SourceIdentity

Gunakan kunci ini untuk membandingkan identitas sumber yang ditetapkan oleh prinsipal dengan identitas sumber yang Anda tentukan dalam kebijakan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setelah identitas sumber ditetapkan saat peran diasumsikan menggunakan CLI perintah, atau AWS STS operasi peran apa pun. AWS STS `AssumeRole` API
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci ini dalam kebijakan untuk mengizinkan tindakan AWS berdasarkan prinsipal yang telah menetapkan identitas sumber saat mengambil peran. Aktivitas untuk identitas sumber yang ditentukan peran muncul diAWS CloudTrail. Hal ini memudahkan administrator untuk menentukan siapa atau apa yang melakukan tindakan dengan peran di dalamnya AWS.

Tidak seperti `sts:RoleSessionName`, setelah identitas sumber diatur, nilai tidak dapat diubah. Ini terdapat dalam konteks permintaan untuk semua tindakan yang diambil oleh peran. Nilai tetap ada dalam sesi peran berikutnya saat Anda menggunakan kredenal sesi untuk mengasumsikan peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

`sts:SourceIdentity`Kuncinya hadir dalam permintaan ketika prinsipal awalnya menetapkan identitas sumber sambil mengasumsikan peran menggunakan perintah peran apa pun AWS STS , atau operasiCLI. AWS STS `AssumeRole` API Kunci `aws:SourceIdentity` tersedia dalam permintaan untuk tindakan apa pun yang diambil dengan sesi peran yang menetapkan identitas sumber.

Kebijakan kepercayaan peran berikut untuk `CriticalRole` dalam akun `111122223333` berisi kondisi untuk `aws:SourceIdentity` yang mencegah prinsipal tanpa identitas sumber yang diatur ke Saanvi atau Diego dari mengasumsikan peran.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya tentang menggunakan informasi identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

EC2: RoleDelivery

Gunakan kunci ini untuk membandingkan versi layanan metadata instans dalam permintaan yang ditandatangani dengan kredenal IAM peran untuk Amazon. EC2 Layanan metadata instance membedakan antara IMDSv1 dan IMDSv2 permintaan berdasarkan apakah, untuk setiap permintaan yang diberikan, baik GET header PUT atau, yang unik untukIMDSv2, ada dalam permintaan itu.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali sesi peran dibuat oleh EC2 instans Amazon.
- Tipe data - Numerik
- Jenis nilai - Bernilai tunggal
- Contoh nilai - 1.0, 2.0

Anda dapat mengonfigurasi Layanan Metadata Instance (IMDS) pada setiap instance sehingga kode lokal atau pengguna harus menggunakan. IMDSv2 Ketika Anda menentukan yang IMDSv2 harus digunakan, IMDSv1 tidak lagi berfungsi.

- Layanan Metadata Instance Versi 1 (IMDSv1) — Metode permintaan/respons
- Instance Metadata Service Version 2 (IMDSv2) — metode yang berorientasi pada sesi

Untuk informasi tentang cara mengonfigurasi instans yang akan digunakanIMDSv2, lihat Mengonfigurasi opsi metadata instans.

Dalam contoh berikut, akses ditolak jika RoleDelivery nilai ec2: dalam konteks permintaan adalah 1.0 (IMDSv1). Pernyataan kebijakan ini dapat diterapkan secara umum karena, jika permintaan tidak ditandatangani oleh kredensi EC2 peran Amazon, maka tidak akan berpengaruh.

```
{
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
```

```
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Contoh kebijakan untuk bekerja dengan metadata instance.

EC2: SourceInstanceArn

Gunakan kunci ini untuk membandingkan contoh dari mana sesi peran dihasilkan. ARN

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali sesi peran dibuat oleh EC2 instans Amazon.
- Tipe data - ARN
- Jenis nilai - Bernilai tunggal
- Nilai contoh — arn:aws: :ec2:us-west- 2:111111111111: instance/instance-id

Untuk contoh kebijakan, lihat Mengizinkan instance tertentu untuk melihat sumber daya di AWS layanan lain.

lem: RoleAssumedBy

AWS Glue Layanan menetapkan kunci kondisi ini untuk setiap AWS API permintaan di mana AWS Glue membuat permintaan menggunakan peran layanan atas nama pelanggan (bukan oleh pekerjaan atau titik akhir pengembang, tetapi langsung oleh AWS Glue layanan). Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari AWS Glue layanan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat AWS Glue membuat permintaan menggunakan peran layanan atas nama pelanggan.
- Tipe data - String
- Jenis nilai - Bernilai tunggal
- Nilai contoh - Kunci ini selalu diatur `keglue.amazonaws.com`.

Contoh berikut menambahkan kondisi untuk memungkinkan AWS Glue layanan mendapatkan objek dari bucket Amazon S3.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
    }
}
```

lem: CredentialIssuingService

AWS Glue Layanan menetapkan kunci ini untuk setiap AWS API permintaan menggunakan peran layanan yang berasal dari pekerjaan atau titik akhir pengembang. Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari AWS Glue pekerjaan atau titik akhir pengembang.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat AWS Glue membuat permintaan yang berasal dari pekerjaan atau titik akhir pengembang.
- Tipe data - String
- Jenis nilai - Bernilai tunggal
- Nilai contoh - Kunci ini selalu diatur ke`glue.amazonaws.com`.

Contoh berikut menambahkan kondisi yang melekat pada IAM peran yang digunakan oleh AWS Glue pekerjaan. Ini memastikan tindakan tertentu diizinkan/ditolak berdasarkan apakah sesi peran digunakan untuk lingkungan runtime AWS Glue pekerjaan.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

lambda: SourceFunctionArn

Gunakan kunci ini untuk mengidentifikasi fungsi Lambda tempat ARN kredensil IAM peran dikirimkan. Layanan Lambda menetapkan kunci ini untuk setiap AWS API permintaan yang berasal dari lingkungan eksekusi fungsi Anda. Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari kode fungsi Lambda tertentu. Lambda juga menetapkan kunci ini untuk beberapa permintaan yang datang dari luar lingkungan eksekusi, seperti menulis log ke CloudWatch dan mengirim jejak ke X-Ray.

- Ketersediaan - Kunci ini disertakan dalam konteks permintaan setiap kali kode fungsi Lambda dipanggil.
- Tipe data - ARN
- Jenis nilai - Bernilai tunggal
- Nilai contoh — arn:aws: :lambda: us-east- 1:123456789012: fungsi: TestFunction

Contoh berikut memungkinkan satu fungsi Lambda tertentu untuk `s3:PutObject` mengakses bucket yang ditentukan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws::lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Bekerja dengan kredenal lingkungan eksekusi Lambda di Panduan Pengembang.AWS Lambda

ssm: SourceInstanceArn

Gunakan kunci ini untuk mengidentifikasi instance AWS Systems Manager terkelola ARN yang dikirimkan ke kredensil IAM peran. Kunci kondisi ini tidak ada saat permintaan berasal dari instance terkelola dengan IAM peran yang terkait dengan profil EC2 instans Amazon.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali kredensil peran dikirimkan ke instance AWS Systems Manager terkelola.
- Tipe data - ARN
- Jenis nilai - Bernilai tunggal
- Nilai contoh — arn:aws: :ec2:us-west- 2:111111111111: instance/instance-id

toko identitas: UserId

Gunakan kunci ini untuk membandingkan IAM identitas tenaga kerja Pusat Identitas dalam permintaan yang ditandatangani dengan identitas yang ditentukan dalam kebijakan.

- Ketersediaan — Kunci ini disertakan ketika pemanggil permintaan adalah pengguna di Pusat IAM Identitas.
- Tipe data - String
- Jenis nilai - Bernilai tunggal
- Nilai contoh — 94482488-3041-7026-18f3-be45837cd0e4

Anda dapat menemukan pengguna di Pusat IAM Identitas dengan membuat permintaan untuk GetUserIdAPImenggunakan AWS CLI, AWS API, atau AWS SDK. UserId

Properti jaringan

Gunakan tombol kondisi berikut untuk membandingkan detail tentang jaringan tempat permintaan berasal atau diteruskan dengan properti jaringan yang Anda tentukan dalam kebijakan.

Daftar Isi

- aws: VpcSourceIp

aws: SourceIp

Gunakan kunci ini untuk membandingkan alamat IP pemohon dengan alamat IP yang Anda tentukan dalam kebijakan. Kunci kondisi `aws:SourceIp` hanya dapat digunakan untuk rentang alamat IP publik.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan, kecuali saat pemohon menggunakan VPC titik akhir untuk membuat permintaan.
- Tipe data — alamat IP
- Jenis nilai - Bernilai tunggal

Kunci kondisi `aws:SourceIp` dapat digunakan dalam kebijakan yang memungkinkan prinsipal hanya membuat permintaan dari rentang IP yang ditetapkan.

> **Note**
>
> `aws:SourceIp`mendukung keduanya IPv4 dan IPv6 alamat atau rentang alamat IP. Untuk daftar dukungan Layanan AWS tersebutIPv6, lihat dukungan Layanan AWS tersebut IPv6 di Panduan VPC Pengguna Amazon.

Misalnya, Anda dapat melampirkan kebijakan berbasis identitas berikut ke peran. IAM Kebijakan ini memungkinkan pengguna untuk memasukkan objek ke dalam bucket `amzn-s3-demo-bucket3` Amazon S3 jika mereka melakukan panggilan dari rentang IPv4 alamat yang ditentukan. Kebijakan ini juga memungkinkan AWS layanan yang digunakan Teruskan sesi akses untuk melakukan operasi ini atas nama Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
```

```
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        }
    ]
}
```

Jika Anda perlu membatasi akses dari jaringan yang mendukung keduanya IPv4 dan IPv6 pengalamatan, Anda dapat menyertakan IPv4 dan IPv6 alamat atau rentang alamat IP dalam kondisi IAM kebijakan. Kebijakan berbasis identitas berikut akan memungkinkan pengguna untuk memasukkan objek ke dalam bucket `amzn-s3-demo-bucket3` Amazon S3 jika pengguna melakukan panggilan dari rentang yang ditentukan atau alamat. IPv4 IPv6 Sebelum Anda memasukkan rentang IPv6 alamat dalam IAM kebijakan Anda, verifikasi bahwa Layanan AWS Anda bekerja dengan dukunganIPv6. Untuk daftar dukungan Layanan AWS tersebutIPv6, lihat dukungan Layanan AWS tersebut IPv6 di Panduan VPC Pengguna Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

Jika permintaan berasal dari host yang menggunakan VPC endpoint Amazon, maka `aws:SourceIp` kuncinya tidak tersedia. Sebagai gantinya, Anda harus menggunakan kunci VPC -specific seperti aws: VpcSourceIp. Untuk informasi selengkapnya tentang penggunaan VPC titik akhir, lihat Identitas dan manajemen akses untuk layanan VPC titik akhir dan VPC titik akhir di Panduan.AWS PrivateLink

aws: SourceVpc

Gunakan kunci ini untuk memeriksa apakah permintaan berjalan melalui VPC titik akhir VPC yang dilampirkan. Dalam kebijakan, Anda dapat menggunakan kunci ini untuk mengizinkan akses hanya ke yang spesifikVPC. Untuk informasi selengkapnya, lihat Membatasi Akses ke Spesifik VPC di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika pemohon menggunakan VPC titik akhir untuk membuat permintaan.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Dalam kebijakan, Anda dapat menggunakan kunci ini untuk mengizinkan atau membatasi akses ke yang spesifikVPC.

Misalnya, Anda dapat melampirkan kebijakan berbasis identitas berikut ke IAM peran yang akan ditolak `PutObject` ke bucket `amzn-s3-demo-bucket3` Amazon S3, kecuali permintaan dibuat dari VPC ID yang ditentukan atau dengan Layanan AWS itu gunakan sesi akses teruskan (FAS) untuk membuat permintaan atas nama peran. Berbeda denganaws: SourceIp, Anda harus menggunakan AWS: V iaAWSService atau aws: CalledVia mengizinkan FAS permintaan, karena VPC sumber permintaan awal tidak dipertahankan.

> **Note**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PutObjectIfNotVPCID",
      "Effect": "Deny",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-1234567890abcdef0"
```

```
        },
        "Bool": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Untuk contoh cara menerapkan kunci ini dalam kebijakan berbasis sumber daya, lihat Membatasi akses ke spesifik di VPC Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

aws: SourceVpce

Gunakan kunci ini untuk membandingkan pengenal VPC titik akhir permintaan dengan ID titik akhir yang Anda tentukan dalam kebijakan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika pemohon menggunakan VPC titik akhir untuk membuat permintaan.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Dalam kebijakan, Anda dapat menggunakan kunci ini untuk membatasi akses ke titik VPC akhir tertentu. Untuk informasi selengkapnya, lihat Membatasi akses ke yang spesifik VPC di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon. Sama halnya dengan menggunakanaws: SourceVpc, Anda harus menggunakan AWS: V iaAWSService atau aws: CalledVia mengizinkan permintaan yang dibuat dengan Layanan AWS menggunakan sesi akses teruskan (FAS). Ini karena VPC titik akhir sumber permintaan awal tidak dipertahankan.

aws: VpcSourceIp

Gunakan kunci ini untuk membandingkan alamat IP tempat permintaan dibuat dengan alamat IP yang Anda tentukan dalam kebijakan. Dalam kebijakan, kunci hanya cocok jika permintaan berasal dari alamat IP yang ditentukan dan melewati titik VPC akhir.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika permintaan dibuat menggunakan VPC titik akhir.
- Tipe data — alamat IP
- Jenis nilai - Bernilai tunggal

Untuk informasi selengkapnya, lihat Mengontrol akses ke VPC titik akhir menggunakan kebijakan titik akhir di VPCPanduan Pengguna Amazon. Sama halnya dengan menggunakanaws: SourceVpc, Anda harus menggunakan AWS: V iaAWSService atau aws: CalledVia mengizinkan permintaan yang dibuat dengan Layanan AWS menggunakan sesi akses teruskan (FAS). Ini karena IP sumber dari permintaan awal yang dibuat menggunakan VPC titik akhir tidak disimpan dalam FAS permintaan.

> **Note**
>
> `aws:VpcSourceIp`mendukung keduanya IPv4 dan IPv6 alamat atau rentang alamat IP. Untuk daftar dukungan Layanan AWS tersebutIPv6, lihat dukungan Layanan AWS tersebut IPv6 di Panduan VPC Pengguna Amazon.
> Kunci `aws:VpcSourceIp` kondisi harus selalu digunakan bersama dengan salah satu `aws:SourceVpc` atau tombol `aws:SourceVpce` kondisi. Jika tidak, ada kemungkinan API panggilan dari yang tidak terduga VPC yang menggunakan IP CIDR yang sama atau tumpang tindih diizinkan oleh kebijakan. Hal ini dapat terjadi karena IP CIDRs dari dua yang tidak terkait VPCs bisa sama atau tumpang tindih. Sebaliknya, VPC IDs atau VPC Endpoint IDs harus digunakan dalam kebijakan karena mereka memiliki pengidentifikasi unik secara global. Pengidentifikasi unik ini memastikan bahwa hasil yang tidak terduga tidak akan terjadi.

Properti dari sumber daya

Gunakan tombol kondisi berikut untuk membandingkan detail tentang sumber daya yang menjadi target permintaan dengan properti sumber daya yang Anda tentukan dalam kebijakan.

Daftar Isi

aws: ResourceAccount

Gunakan kunci ini untuk membandingkan Akun AWS ID pemilik sumber daya yang diminta dengan akun sumber daya dalam kebijakan. Anda kemudian dapat mengizinkan atau menolak akses ke sumber daya tersebut berdasarkan akun yang memiliki sumber daya tersebut.

- Ketersediaan — Kunci ini selalu disertakan dalam konteks permintaan untuk sebagian besar tindakan layanan. Tindakan berikut tidak mendukung kunci ini:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store - Semua tindakan
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CopyImage`
    - `ec2:CopySnapshot`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVolume`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
    - `ec2:DeleteTransitGatewayPeeringAttachment`
    - `ec2:DeleteVpcPeeringConnection`
    - `ec2:RejectTransitGatewayPeeringAttachment`
    - `ec2:RejectVpcEndpointConnections`
    - `ec2:RejectVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents`— EventBridge `PutEvents` memanggil bus acara di akun lain, jika bus acara itu dikonfigurasi sebagai EventBridge target lintas akun sebelum 2 Maret 2023. Untuk informasi selengkapnya, lihat Memberikan izin untuk mengizinkan peristiwa dari AWS akun lain di Panduan EventBridge Pengguna Amazon.
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`

  - OpenSearch Layanan Amazon
    - `es:AcceptInboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
    - `route53:DeleteVPCAssociationAuthorization`
    - `route53:DisassociateVPCFromHostedZone`
    - `route53:ListHostedZonesByVPC`
  - AWS Security Hub
    - `securityhub:AcceptAdministratorInvitation`
- Tipe data - String
- Jenis nilai - Bernilai tunggal

> **Note**
>
> Untuk pertimbangan tambahan untuk tindakan yang tidak didukung di atas, lihat repositori Contoh Kebijakan Perimeter Data.

Kunci ini sama dengan Akun AWS ID untuk akun dengan sumber daya yang dievaluasi dalam permintaan.

Untuk sebagian besar sumber daya di akun Anda, ARNberisi ID akun pemilik untuk sumber daya tersebut. Untuk sumber daya tertentu, seperti bucket Amazon S3, sumber daya ARN tidak menyertakan ID akun. Dua contoh berikut menunjukkan perbedaan antara sumber daya dengan ID akun diARN, dan Amazon S3 ARN tanpa ID akun:

- `arn:aws:iam::123456789012:role/AWSExampleRole`— IAM peran dibuat dan dimiliki dalam akun 123456789012.
- `arn:aws:s3:::amzn-s3-demo-bucket2`- Bucket Amazon S3 dibuat dan dimiliki di dalam akun`111122223333`, tidak ditampilkan di. ARN

Gunakan AWS konsol, atau, atau APICLI, untuk menemukan semua sumber daya Anda dan yang sesuaiARNs.

Anda menulis kebijakan yang menolak izin ke sumber daya berdasarkan ID akun pemilik sumber daya. Misalnya, kebijakan berbasis identitas berikut menolak akses ke sumber daya yang ditentukan jika sumber daya bukan milik akun yang ditentukan.

Untuk menggunakan kebijakan ini, ganti teks placeholder yang dicetak miring dengan informasi akun Anda.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Sebaliknya, ia menggunakan `Deny` efek yang secara eksplisit menolak akses ke semua sumber daya yang tercantum dalam pernyataan yang bukan milik akun yang terdaftar. Gunakan kebijakan ini dalam kombinasi dengan kebijakan lain yang memungkinkan akses ke sumber daya tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:account:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "account"
          ]
        }
      }
    }
  ]
}
```

Kebijakan ini menolak akses ke semua sumber daya untuk AWS layanan tertentu kecuali yang ditentukan Akun AWS memiliki sumber daya.

> **Note**
>
> Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang di-host di tempat lain Akun AWS. Penggunaan `aws:ResourceAccount` dalam kebijakan berbasis identitas Anda dapat memengaruhi kemampuan identitas Anda untuk mengakses sumber daya ini.

AWS Layanan tertentu, seperti AWS Data Exchange, mengandalkan akses ke sumber daya di luar Anda Akun AWS untuk operasi normal. Jika Anda menggunakan elemen `aws:ResourceAccount` dalam kebijakan Anda, sertakan pernyataan tambahan untuk membuat pengecualian untuk layanan tersebut. Kebijakan contoh AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange menunjukkan cara menolak akses berdasarkan akun sumber daya sambil menentukan pengecualian untuk sumber daya milik layanan.

Gunakan contoh kebijakan ini sebagai templat untuk membuat kebijakan kustom Anda sendiri. Lihat dokumentasi layanan Anda untuk informasi lebih lanjut.

aws: ResourceOrgPaths

Gunakan kunci ini untuk membandingkan jalur AWS Organizations untuk sumber daya yang diakses dengan jalur dalam kebijakan. Dalam kebijakan, kunci kondisi ini memastikan bahwa sumber daya milik anggota akun dalam akar organisasi tertentu atau unit organisasi (OUs) di AWS Organizations. AWS Organizations path adalah representasi teks dari struktur entitas Organizations. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat Memahami jalur entitas AWS Organizations

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika akun yang memiliki sumber daya adalah anggota organisasi. Kunci kondisi global ini tidak mendukung tindakan berikut:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store - Semua tindakan
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`

  - `ec2:AcceptVpcPeeringConnection`
  - `ec2:CopyImage`
  - `ec2:CopySnapshot`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVolume`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
  - `ec2:DeleteTransitGatewayPeeringAttachment`
  - `ec2:DeleteVpcPeeringConnection`
  - `ec2:RejectTransitGatewayPeeringAttachment`
  - `ec2:RejectVpcEndpointConnections`
  - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` memanggil bus acara di akun lain, jika bus acara itu dikonfigurasi sebagai EventBridge target lintas akun sebelum 2 Maret 2023. Untuk informasi selengkapnya, lihat Memberikan izin untuk mengizinkan peristiwa dari AWS akun lain di Panduan EventBridge Pengguna Amazon.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Layanan Amazon
  - `es:AcceptInboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`
  - `route53:ListHostedZonesByVPC`
- AWS Security Hub
  - `securityhub:AcceptAdministratorInvitation`

- Tipe data - String (daftar)
- Jenis nilai — Multivalued

> **Note**
>
> Untuk pertimbangan tambahan untuk tindakan yang tidak didukung di atas, lihat repositori Contoh Kebijakan Perimeter Data.

`aws:ResourceOrgPaths` adalah kunci kondisi multinilai. Kunci multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan `ForAnyValue` atau `ForAllValues` mengatur operator dengan operator kondisi string untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Kunci konteks multivaluasi.

Misalnya, kondisi berikut mengembalikan `True` sumber daya milik organisasio`-a1b2c3d4e5`. Ketika Anda menyertakan wildcard, Anda harus menggunakan operator StringLikekondisi.

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

Kondisi berikut mengembalikan `True` sumber daya dengan ID OU`ou-ab12-11111111`. Ini akan cocok dengan sumber daya yang dimiliki oleh akun yang dilampirkan pada OU ou-ab12-11111111 atau salah satu anak. OUs

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

Kondisi berikut mengembalikan sumber daya `True` yang dimiliki oleh akun yang dilampirkan langsung ke ID OU`ou-ab12-22222222`, tetapi bukan anakOUs. Contoh berikut menggunakan operator StringEqualskondisi untuk menentukan persyaratan kecocokan yang tepat untuk ID OU dan bukan pencocokan wildcard.

```
"Condition": { "ForAnyValue:StringEquals" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
```

```
}}
```

> **Note**
>
> Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang di-host di tempat lain Akun AWS. Penggunaan `aws:ResourceOrgPaths` dalam kebijakan berbasis identitas Anda dapat memengaruhi kemampuan identitas Anda untuk mengakses sumber daya ini.

AWS Layanan tertentu, seperti AWS Data Exchange, mengandalkan akses ke sumber daya di luar Anda Akun AWS untuk operasi normal. Jika Anda menggunakan `aws:ResourceOrgPaths` kunci dalam kebijakan Anda, sertakan pernyataan tambahan untuk membuat pengecualian untuk layanan tersebut. Kebijakan contoh AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange menunjukkan cara menolak akses berdasarkan akun sumber daya sambil menentukan pengecualian untuk sumber daya milik layanan. Anda dapat membuat kebijakan serupa untuk membatasi akses ke sumber daya dalam unit organisasi (OU) menggunakan `aws:ResourceOrgPaths` kunci, sambil memperhitungkan sumber daya milik layanan.

Gunakan contoh kebijakan ini sebagai templat untuk membuat kebijakan kustom Anda sendiri. Lihat dokumentasi layanan Anda untuk informasi lebih lanjut.

aws: ResourceOrg ID

Gunakan kunci ini untuk membandingkan pengenal AWS organisasi dalam Organizations yang menjadi sumber daya yang diminta dengan pengenal yang ditentukan dalam kebijakan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika akun yang memiliki sumber daya adalah anggota organisasi. Kunci kondisi global ini tidak mendukung tindakan berikut:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store - Semua tindakan
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`

  - `ec2:AcceptVpcPeeringConnection`
  - `ec2:CopyImage`
  - `ec2:CopySnapshot`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVolume`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
  - `ec2:DeleteTransitGatewayPeeringAttachment`
  - `ec2:DeleteVpcPeeringConnection`
  - `ec2:RejectTransitGatewayPeeringAttachment`
  - `ec2:RejectVpcEndpointConnections`
  - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` memanggil bus acara di akun lain, jika bus acara itu dikonfigurasi sebagai EventBridge target lintas akun sebelum 2 Maret 2023. Untuk informasi selengkapnya, lihat Memberikan izin untuk mengizinkan peristiwa dari AWS akun lain di Panduan EventBridge Pengguna Amazon.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Layanan Amazon
  - `es:AcceptInboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`
  - `route53:ListHostedZonesByVPC`
- AWS Security Hub
  - `securityhub:AcceptAdministratorInvitation`

- Tipe data - String
- Jenis nilai - Bernilai tunggal

> **Note**
>
> Untuk pertimbangan tambahan untuk tindakan yang tidak didukung di atas, lihat repositori Contoh Kebijakan Perimeter Data.

Kunci global ini mengembalikan ID organisasi sumber daya untuk permintaan yang diberikan. Ini memungkinkan Anda untuk membuat aturan yang berlaku untuk semua sumber daya dalam organisasi yang ditentukan dalam `Resource` elemen kebijakan berbasis identitas. Anda dapat menentukan ID organisasi dalam elemen kondisi. Saat Anda menambahkan dan menghapus akun, kebijakan yang menyertakan `aws:ResourceOrgID` kunci secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbaruinya secara manual.

Misalnya, kebijakan berikut mencegah prinsipal menambahkan objek ke `policy-genius-dev` sumber daya kecuali sumber daya Amazon S3 milik organisasi yang sama dengan prinsipal yang membuat permintaan.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Sebaliknya, ia menggunakan `Deny` efek yang secara eksplisit menolak akses ke semua sumber daya yang tercantum dalam pernyataan yang bukan milik akun yang terdaftar. Gunakan kebijakan ini dalam kombinasi dengan kebijakan lain yang memungkinkan akses ke sumber daya tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
    "Effect": "Deny",
    "Action": "s3:PutObject",
    "Resource": "arn:partition:s3:::policy-genius-dev/*",
    "Condition": {
      "StringNotEquals": {
        "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
```

```
      }
    }
  }
}
```

> **Note**
>
> Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang di-host di tempat lain Akun AWS. Penggunaan `aws:ResourceOrgID` dalam kebijakan berbasis identitas Anda dapat memengaruhi kemampuan identitas Anda untuk mengakses sumber daya ini.

AWS Layanan tertentu, seperti AWS Data Exchange, mengandalkan akses ke sumber daya di luar Anda Akun AWS untuk operasi normal. Jika Anda menggunakan `aws:ResourceOrgID` kunci dalam kebijakan Anda, sertakan pernyataan tambahan untuk membuat pengecualian untuk layanan tersebut. Kebijakan contoh AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange menunjukkan cara menolak akses berdasarkan akun sumber daya sambil menentukan pengecualian untuk sumber daya milik layanan. Anda dapat membuat kebijakan serupa untuk membatasi akses ke sumber daya dalam organisasi Anda menggunakan `aws:ResourceOrgID` kunci, sambil memperhitungkan sumber daya milik layanan.

Gunakan contoh kebijakan ini sebagai templat untuk membuat kebijakan kustom Anda sendiri. Lihat dokumentasi layanan Anda untuk informasi lebih lanjut.

Dalam video berikut, pelajari selengkapnya tentang cara menggunakan kunci kondisi `aws:ResourceOrgID` dalam kebijakan.

Pastikan identitas dan jaringan hanya dapat digunakan untuk mengakses sumber daya tepercaya.

aws:ResourceTag/tag-kunci

Gunakan kunci ini untuk membandingkan pasangan nilai kunci tag yang Anda tentukan dalam kebijakan dengan pasangan nilai kunci yang dilampirkan ke sumber daya. Misalnya, Anda dapat meminta agar akses ke sumber daya hanya diperbolehkan jika sumber daya memiliki kunci tanda yang dilampirkan `"Dept"` dengan nilai `"Marketing"`. Untuk informasi selengkapnya, lihat Mengontrol akses ke sumber daya AWS.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan ketika sumber daya yang diminta sudah memiliki tag terlampir atau dalam permintaan yang membuat sumber daya dengan tag

terlampir. Kunci ini hanya dikembalikan untuk sumber daya yang mendukung otorisasi berdasarkan tanda. Ada satu kunci konteks untuk setiap pasangan nilai kunci tanda.

- Tipe data - String
- Jenis nilai - Bernilai tunggal

Kunci konteks ini diformat `"aws:ResourceTag/tag-key":"tag-value"` jika *`tag-key`* and *`tag-value`* adalah kunci tag dan pasangan nilai. Kunci dan nilai tag tidak peka huruf besar/kecil. Ini berarti jika Anda menentukan `"aws:ResourceTag/TagKey1": "Value1"` dalam elemen ketentuan kebijakan Anda, kemudian ketentuan tersebut cocok dengan kunci tanda sumber daya bernama `TagKey1` atau `tagkey1`, tetapi tidak keduanya.

Untuk contoh menggunakan `aws:ResourceTag` kunci untuk mengontrol akses ke IAM sumber daya, lihatMengontrol akses ke sumber daya AWS.

Untuk contoh menggunakan `aws:ResourceTag` kunci untuk mengontrol akses ke AWS sumber daya lain, lihatMengontrol akses ke AWS sumber daya menggunakan tag.

Untuk tutorial tentang menggunakan kunci `aws:ResourceTag` kondisi untuk atribut based access control (ABAC), lihatIAMtutorial: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag.

Properti permintaan

Gunakan tombol kondisi berikut untuk membandingkan detail tentang permintaan itu sendiri dan konten permintaan dengan properti permintaan yang Anda tentukan dalam kebijakan.

Daftar Isi

- aws: SecureTransport
- aws: SourceArn
- aws: SourceAccount
- aws: SourceOrgPaths
- aws: SourceOrg ID
- aws: UserAgent

aws: CalledVia

Gunakan kunci ini untuk membandingkan layanan dalam kebijakan dengan layanan yang membuat permintaan atas nama IAM prinsipal (pengguna atau peran). Ketika kepala sekolah membuat permintaan ke AWS layanan, layanan itu mungkin menggunakan kredensi kepala sekolah untuk membuat permintaan berikutnya ke layanan lain. Kunci `aws:CalledVia` berisi daftar yang dipesan dari setiap layanan dalam rantai yang membuat permintaan atas nama utama.

Misalnya, Anda dapat menggunakan AWS CloudFormation untuk membaca dan menulis dari tabel Amazon DynamoDB. DynamoDB kemudian menggunakan enkripsi yang disediakan AWS Key Management Service oleh ().AWS KMS

- Ketersediaan — Kunci ini hadir dalam permintaan ketika layanan yang mendukung `aws:CalledVia` menggunakan kredensi IAM prinsipal untuk mengajukan permintaan ke layanan lain. Kunci ini tidak ada jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data - String (daftar)
- Jenis nilai — Multivalued

Untuk menggunakan kunci `aws:CalledVia` kondisi dalam kebijakan, Anda harus memberikan prinsip layanan untuk mengizinkan atau menolak AWS permintaan layanan. AWS mendukung menggunakan prinsip layanan berikut dengan. `aws:CalledVia`

Pemimpin layanan
aoss.amazonaws.com

Pemimpin layanan
athena.amazonaws.com
backup.amazonaws.com
cloud9.amazonaws.com
cloudformation.amazonaws.com
databrew.amazonaws.com
dataexchange.amazonaws.com
dynamodb.amazonaws.com
imagebuilder.amazonaws.com
kms.amazonaws.com
mgn.amazonaws.com
nimble.amazonaws.com
omics.amazonaws.com
ram.amazonaws.com
robomaker.amazonaws.com
servicecatalog-appregistry.amazonaws.com
sqlworkbench.amazonaws.com
ssm-guiconnect.amazonaws.com

Untuk memungkinkan atau menolak akses ketika setiap layanan membuat permintaan dengan menggunakan kredensial prinsipal, gunakan kunci kondisi `AWS: V iaAWSService`. Kunci kondisi itu mendukung AWS layanan.

Kunci `aws:CalledVia` adalah tombol multinilai. Namun, Anda tidak dapat menerapkan perintah menggunakan kunci ini dalam suatu kondisi. Menggunakan contoh di atas, Pengguna 1 membuat permintaan ke AWS CloudFormation, yang memanggil DynamoDB, yang memanggil. AWS KMS Ini adalah tiga permintaan terpisah. Panggilan terakhir ke AWS KMS dilakukan oleh Pengguna 1 melalui AWS CloudFormation dan kemudian DynamoDB.

Dalam kasus ini, kunci `aws:CalledVia` dalam konteks permintaan mencakup `cloudformation.amazonaws.com` dan `dynamodb.amazonaws.com`, dalam urutan tersebut. Jika Anda hanya peduli bahwa panggilan dilakukan melalui DynamoDB di suatu tempat dalam rantai permintaan, Anda dapat menggunakan kunci kondisi ini dalam kebijakan Anda.

Misalnya, kebijakan berikut memungkinkan pengelolaan AWS KMS kunci bernama`my-example-key`, tetapi hanya jika DynamoDB adalah salah satu layanan yang meminta. Operator `ForAnyValue:StringEquals` kondisi memastikan bahwa DynamoDB adalah salah satu layanan panggilan. Jika prinsipal langsung memanggil AWS KMS , kondisi akan memberikan `false` dan permintaan tersebut tidak diizinkan oleh kebijakan ini.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": ["dynamodb.amazonaws.com"]
                }
            }
        }
    ]
}
```

Jika Anda ingin memberlakukan layanan mana yang membuat panggilan pertama atau terakhir dalam rantai, Anda dapat menggunakan kunci aws:CalledViaFirst dan aws:CalledViaLast. Misalnya, kebijakan berikut memungkinkan pengelolaan kunci yang diberi nama my-example-key AWS KMS. AWS KMS Operasi ini hanya diperbolehkan jika beberapa permintaan disertakan dalam rantai. Permintaan pertama harus dilakukan melalui AWS CloudFormation dan yang terakhir melalui DynamoDB. Jika layanan lain membuat permintaan di tengah rantai, operasi masih diizinkan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
}
```

aws:CalledViaLastKunci aws:CalledViaFirst dan hadir dalam permintaan ketika layanan menggunakan kredensi IAM kepala sekolah untuk memanggil layanan lain. Mereka menunjukkan layanan pertama dan terakhir yang melakukan panggilan dalam rantai permintaan. Misalnya, asumsikan bahwa AWS CloudFormation memanggil layanan lain bernamaX Service, yang memanggil DynamoDB, yang kemudian memanggil. AWS KMS Panggilan terakhir ke AWS KMS dilakukan oleh User 1 via AWS CloudFormation, thenX Service, dan kemudian DynamoDB. Ini pertama kali dipanggil via AWS CloudFormation dan terakhir dipanggil melalui DynamoDB.

aws: CalledViaFirst

Gunakan kunci ini untuk membandingkan layanan dalam kebijakan dengan layanan pertama yang membuat permintaan atas nama IAM prinsipal (pengguna atau peran). Untuk informasi selengkapnya, lihat `aws:CalledVia`.

- Ketersediaan — Kunci ini hadir dalam permintaan ketika layanan menggunakan kredensi IAM prinsipal untuk membuat setidaknya satu permintaan lain ke layanan yang berbeda. Kunci ini tidak ada jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

aws: CalledViaLast

Gunakan kunci ini untuk membandingkan layanan dalam kebijakan dengan layanan terakhir yang membuat permintaan atas nama IAM prinsipal (pengguna atau peran). Untuk informasi selengkapnya, lihat `aws:CalledVia`.

- Ketersediaan — Kunci ini hadir dalam permintaan ketika layanan menggunakan kredensi IAM prinsipal untuk membuat setidaknya satu permintaan lain ke layanan yang berbeda. Kunci ini tidak ada jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

AWS: V iaAWSService

Gunakan kunci ini untuk memeriksa apakah Layanan AWS membuat permintaan ke layanan lain atas nama Anda menggunakan sesi akses teruskan (FAS).

Kunci konteks permintaan kembali `true` ketika layanan menggunakan sesi akses penerusan untuk membuat permintaan atas nama IAM prinsipal asli. Kunci konteks permintaan juga memberikan `false` saat prisipal langsung memanggil.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.

- Tipe data — Boolean
- Jenis nilai - Bernilai tunggal

aws: CurrentTime

Gunakan kunci ini untuk membandingkan tanggal dan waktu permintaan dengan tanggal dan waktu yang Anda sebutkan dalam kebijakan. Untuk melihat contoh kebijakan yang menggunakan kunci kondisi ini, lihat AWS: Mengizinkan akses berdasarkan tanggal dan waktu.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data - Tanggal
- Jenis nilai - Bernilai tunggal

aws: EpochTime

Gunakan kunci ini untuk membandingkan tanggal dan waktu permintaan dalam epoch atau waktu Unix dengan nilai yang Anda tentukan dalam kebijakan. Kunci ini juga menerima jumlah detik sejak 1 Januari 1970.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data - Tanggal, Numerik
- Jenis nilai - Bernilai tunggal

aws:referer

Gunakan kunci ini untuk membandingkan siapa yang mereferensikan permintaan di browser klien denga perujuk yang Anda tentukan dalam kebijakan. Nilai konteks `aws:referer` permintaan disediakan oleh pemanggil di HTTP header. Header `Referer` disertakan dalam permintaan browser web saat Anda memilih tautan di halaman web. `Referer`Header berisi halaman web tempat tautan dipilih. URL

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika permintaan ke AWS sumber daya dipanggil dengan menautkan dari halaman web URL di browser. Kunci ini tidak disertakan untuk permintaan terprogram karena tidak menggunakan tautan browser untuk mengakses sumber daya AWS .
- Tipe data - String

- Jenis nilai - Bernilai tunggal

Misalnya, Anda dapat mengakses objek Amazon S3 secara langsung menggunakan URL atau menggunakan pemanggilan langsungAPI. Untuk informasi selengkapnya, lihat APIOperasi Amazon S3 secara langsung menggunakan browser web. Saat Anda mengakses objek Amazon S3 dari objek URL yang ada di halaman web, halaman web sumber digunakan. URL `aws:referer` Saat Anda mengakses objek Amazon S3 dengan mengetikkan URL ke browser Anda, tidak `aws:referer` ada. Ketika Anda memanggil secara API langsung, `aws:referer` juga tidak hadir. Anda dapat menggunakan kunci kondisi `aws:referer` dalam kebijakan untuk memungkinkan permintaan yang dibuat dari perujuk spesifik, seperti tautan di halaman web dalam domain perusahaan Anda.

> **Warning**
>
> Kunci ini harus digunakan dengan hati-hati. Menyertakan nilai header perujuk yang diketahui publik bukanlah sesuatu yang aman. Pihak yang tidak berwenang dapat menggunakan browser yang diubah atau disesuaikan untuk menyediakan nilai `aws:referer` yang mereka pilih. Akibatnya, tidak `aws:referer` boleh digunakan untuk mencegah pihak yang tidak berwenang membuat AWS permintaan langsung. Ini ditawarkan untuk memungkinkan pelanggan melindungi konten digital mereka, seperti konten yang disimpan di Amazon S3, agar tidak dirujuk pada pihak ketiga yang tidak berwenang.

aws: RequestedRegion

Gunakan kunci ini untuk membandingkan AWS Wilayah yang dipanggil dalam permintaan dengan Wilayah yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci kondisi global ini untuk mengontrol Wilayah mana yang dapat diminta. Untuk melihat AWS Wilayah untuk setiap layanan, lihat Titik akhir dan kuota Layanan di. Referensi Umum Amazon Web Services

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Beberapa layanan global, sepertiIAM, memiliki titik akhir tunggal. Karena titik akhir ini secara fisik terletak di Wilayah AS Timur (Virginia N.), IAM panggilan selalu dilakukan ke Wilayah us-east-1. Misalnya, jika Anda membuat kebijakan yang menolak akses ke semua layanan jika Wilayah

yang diminta bukan us-west-2, maka panggilan selalu gagal. IAM Untuk melihat contoh cara mengatasinya, lihat NotAction dengan Deny.

> **Note**
>
> Kunci kondisi `aws:RequestedRegion` ini memungkinkan Anda mengontrol titik akhir layanan yang dipilih tetapi tidak mengendalikan dampak operasi. Beberapa layanan memiliki dampak lintas Wilayah..
> Misalnya, Amazon S3 memiliki API operasi yang meluas di seluruh wilayah.
>
> - Anda dapat meminta `s3:PutBucketReplication` di satu Wilayah (yang dipengaruhi oleh kunci kondisi `aws:RequestedRegion`, tetapi Wilayah lainnya dipengaruhi berdasarkan pengaturan konfigurasi replikasi.
> - Anda dapat memanggil `s3:CreateBucket` untuk membuat bucket di wilayah lain, dan menggunakan tombol `s3:LocationConstraint` kondisi untuk mengontrol wilayah yang berlaku.

Anda dapat menggunakan kunci konteks ini untuk membatasi akses ke AWS layanan dalam kumpulan Wilayah tertentu. Misalnya, kebijakan berikut memungkinkan pengguna untuk melihat semua EC2 instans Amazon di AWS Management Console. Namun, hal ini hanya memungkinkan mereka untuk membuat perubahan pada instans di Irlandia (eu-west-1), London (eu-west-2), atau Paris (eu-west-3).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
                "ec2:Run*",
                "ec2:Start*",
                "ec2:Stop*",
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

aws:RequestTag/tag-kunci

Gunakan kunci ini untuk membandingkan pasangan nilai kunci tanda yang diteruskan dalam permintaan dengan pasangan tanda yang Anda sebutkan dalam kebijakan. Misalnya, Anda dapat memeriksa apakah permintaan tersebut menyertakan kunci tanda `"Dept"` dan memiliki nilai `"Accounting"`. Untuk informasi selengkapnya, lihat Mengontrol akses selama permintaan AWS.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat pasangan nilai kunci tag diteruskan dalam permintaan. Ketika beberapa tanda diteruskan dalam permintaan, ada satu kunci konteks untuk setiap pasangan nilai kunci tanda.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

Kunci konteks ini diformat `"aws:RequestTag/`*`tag-key`*`":"`*`tag-value`*`"` jika *`tag-key`* and *`tag-value`* adalah kunci tag dan pasangan nilai. Kunci dan nilai tag tidak peka huruf besar/kecil. Ini

berarti bahwa jika Anda menentukan `"aws:RequestTag/TagKey1": "Value1"` dalam elemen kondisi kebijakan Anda, maka kondisi akan cocok dengan kunci tag permintaan bernama salah satu `TagKey1` atau`tagkey1`, tetapi tidak keduanya.

Contoh ini menunjukkan bahwa meskipun kuncinya bernilai tunggal, Anda masih dapat menggunakan beberapa pasangan kunci-nilai dalam permintaan jika kuncinya berbeda.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:::instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

aws: TagKeys

Gunakan kunci ini untuk membandingkan kunci tanda dalam permintaan dengan kunci yang Anda sebutkan dalam kebijakan. Sebaiknya saat Anda menggunakan kebijakan untuk mengontrol akses menggunakan tag, gunakan tombol `aws:TagKeys` kondisi untuk menentukan kunci tag apa yang diizinkan. Untuk contoh kebijakan dan informasi selengkapnya, lihat the section called “Mengontrol akses berdasarkan kunci tanda”.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan jika operasi mendukung tag yang lewat dalam permintaan.
- Tipe data - String (daftar)
- Jenis nilai — Multivalued

Kunci konteks ini diformat `"aws:TagKeys":"tag-key"` jika *`tag-key`* adalah daftar kunci tag tanpa nilai (misalnya, `["Dept","Cost-Center"]`).

Karena Anda dapat memasukkan beberapa pasangan nilai kunci tanda dalam permintaan, konten permintaan dapat berupa permintaan multinilai. Dalam hal ini, Anda harus menggunakan operator kumpulan `ForAllValues` atau `ForAnyValue`. Untuk informasi selengkapnya, lihat Kunci konteks multivaluasi.

Beberapa layanan mendukung penandaan dengan operasi sumber daya, seperti membuat, mengubah, atau menghapus sumber daya. Untuk memungkinkan penandaan dan operasi sebagai satu panggilan, Anda harus membuat kebijakan yang mencakup tindakan penandaan dan tindakan pemodifikasian sumber daya. Kemudian, Anda dapat menggunakan kunci kondisi `aws:TagKeys` untuk menerapkan penggunaan kunci tanda tertentu dalam permintaan. Misalnya, untuk membatasi tag saat seseorang membuat EC2 snapshot Amazon, Anda harus menyertakan tindakan `ec2:CreateSnapshot` pembuatan dan tindakan `ec2:CreateTags` penandaan dalam kebijakan. Untuk melihat kebijakan skenario yang digunakan ini`aws:TagKeys`, lihat Membuat Snapshot dengan Tag di Panduan EC2 Pengguna Amazon.

aws: SecureTransport

Gunakan kunci ini untuk memeriksa apakah permintaan dikirim menggunakanTLS. Konteks permintaan mengembalikan `true` atau `false`. Dalam kebijakan, Anda dapat mengizinkan tindakan tertentu hanya jika permintaan dikirim menggunakanTLS.

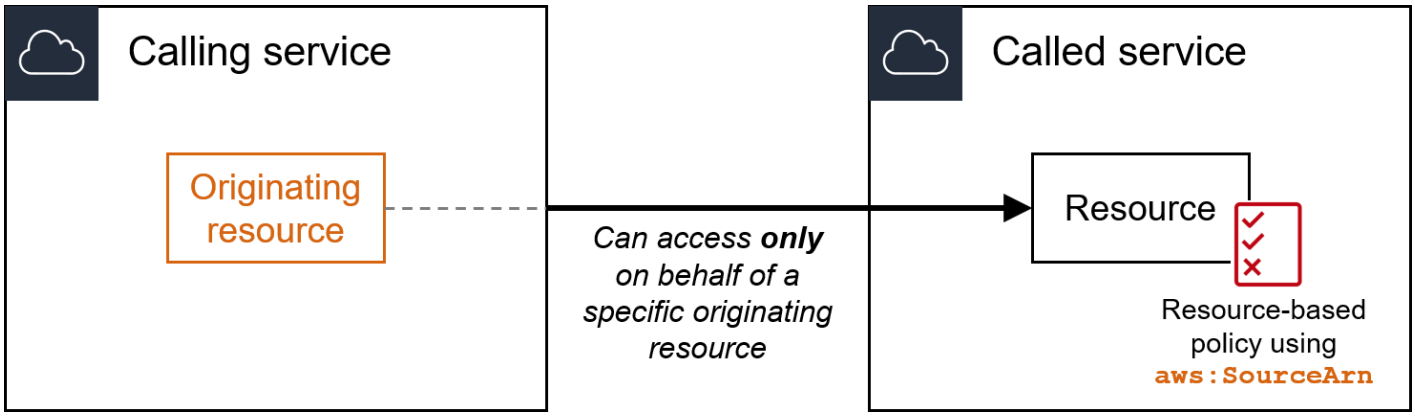
- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — Boolean
- Jenis nilai - Bernilai tunggal

aws: SourceArn

Gunakan kunci ini untuk membandingkan Amazon Resource Name (ARN) sumber daya yang membuat service-to-service permintaan dengan ARN yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh prinsipal AWS layanan. Ketika sumber ARN menyertakan ID akun, tidak perlu digunakan `aws:SourceAccount` dengan`aws:SourceArn`.

Kunci ini tidak bekerja dengan ARN kepala sekolah yang membuat permintaan. Sebaliknya, gunakan `aws: PrincipalArn`.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan langsung oleh kepala AWS layanan atas nama sumber daya yang konfigurasi memicu service-to-service permintaan. Layanan panggilan meneruskan sumber ARN daya asli ke layanan yang disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3

> **Note**
>
> Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceArn` kebijakan KMS kunci untuk kunci yang digunakan oleh Layanan AWS melalui hibah KMS kunci dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data -ARN, String

AWS merekomendasikan agar Anda menggunakan ARNoperator alih-alih operator string saat membandingkanARNs.

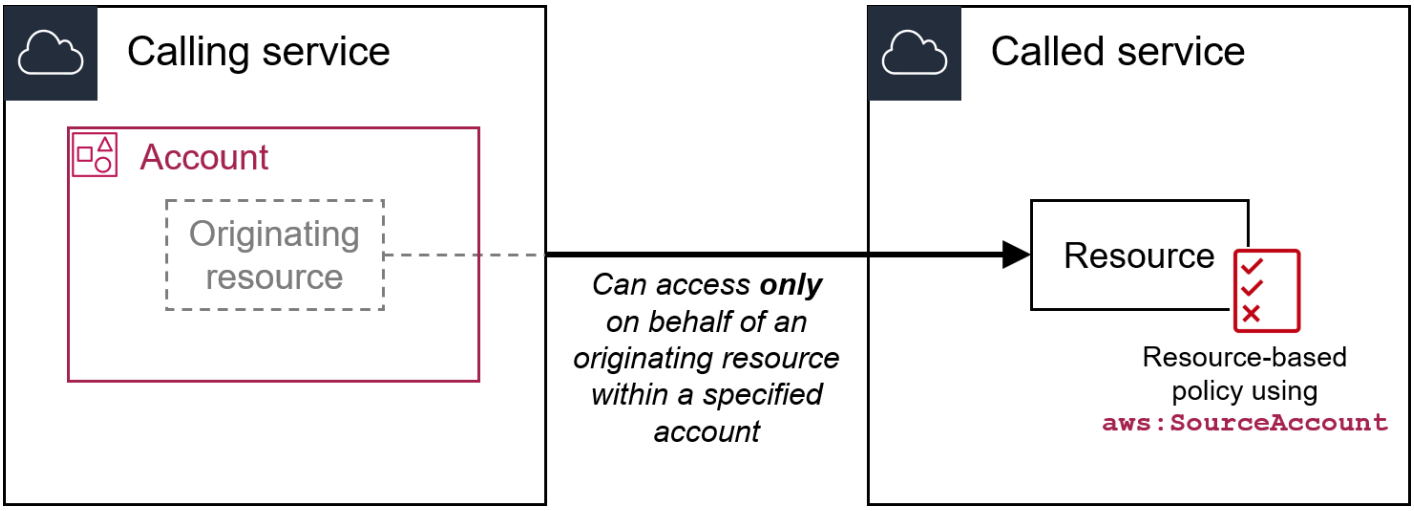
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai wakil yang bingung selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan `Principal` prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke ARN sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi SNS topik Amazon, layanan Amazon S3 akan memanggil operasi. `sns:Publish` API Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke ARN bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

aws: SourceAccount

Gunakan kunci ini untuk membandingkan ID akun sumber daya yang membuat service-to-service permintaan dengan ID akun yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh kepala AWS layanan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan langsung oleh kepala AWS layanan atas nama sumber daya yang konfigurasi memicu service-to-service permintaan. Layanan panggilan meneruskan ID akun dari sumber daya asli ke layanan yang disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3

> **Note**
>
> Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceAccount` kebijakan KMS kunci untuk kunci yang digunakan oleh Layanan AWS melalui hibah KMS kunci dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data - String
- Jenis nilai - Bernilai tunggal

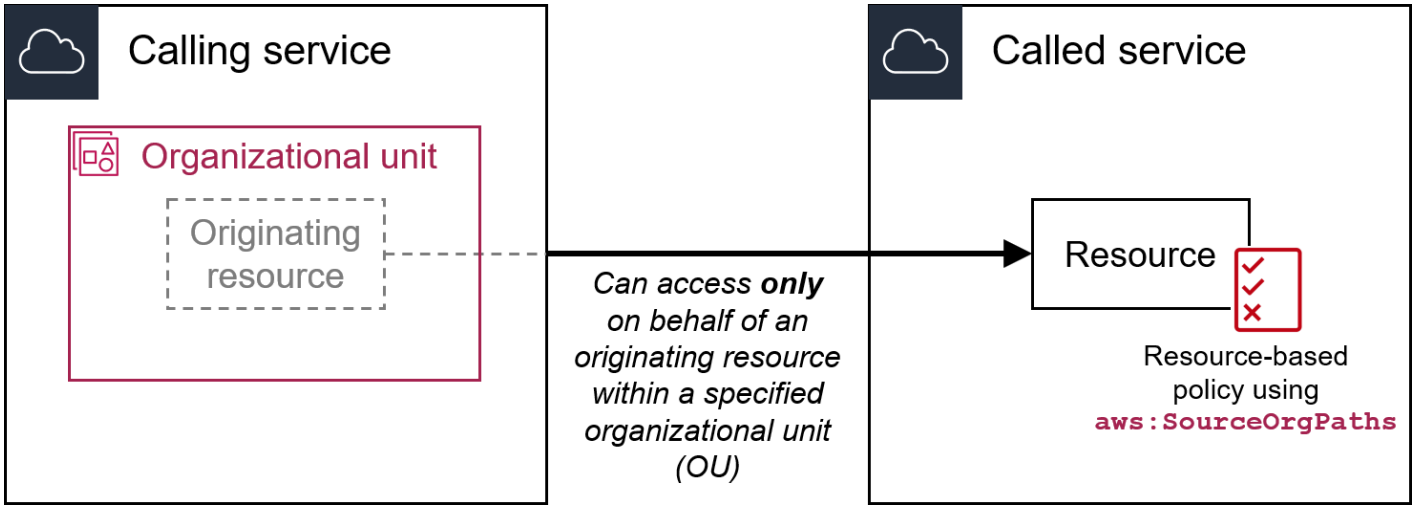
Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai wakil yang bingung selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan `Principal` prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke ID akun sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi SNS topik Amazon, layanan Amazon S3 akan memanggil operasi. `sns:Publish` API Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke ID akun bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

aws: SourceOrgPaths

Gunakan kunci ini untuk membandingkan AWS Organizations jalur sumber daya yang membuat service-to-service permintaan dengan jalur organisasi yang Anda tentukan dalam kebijakan, tetapi

hanya jika permintaan dibuat oleh kepala AWS layanan. Organizations path adalah representasi teks dari struktur entitas Organizations. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat Memahami jalur AWS Organizations entitas.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan langsung oleh kepala AWS layanan atas nama sumber daya yang dimiliki oleh akun yang merupakan anggota organisasi. Layanan panggilan melewati jalur organisasi sumber daya asli ke layanan yang disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3
Semua kepala sekolah layanan	Bot Amazon Lex	Izinkan Layanan AWS untuk menggunakan bot Amazon Lex

> **Note**
>
> Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceOrgPaths` kebijakan KMS kunci untuk kunci yang digunakan oleh Layanan AWS melalui hibah KMS kunci dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data - String (daftar)
- Jenis nilai — Multivalued

Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai wakil yang bingung selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan `Principal` prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke jalur organisasi sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi SNS topik Amazon, layanan Amazon S3 akan memanggil operasi. `sns:Publish` API Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke jalur organisasi bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

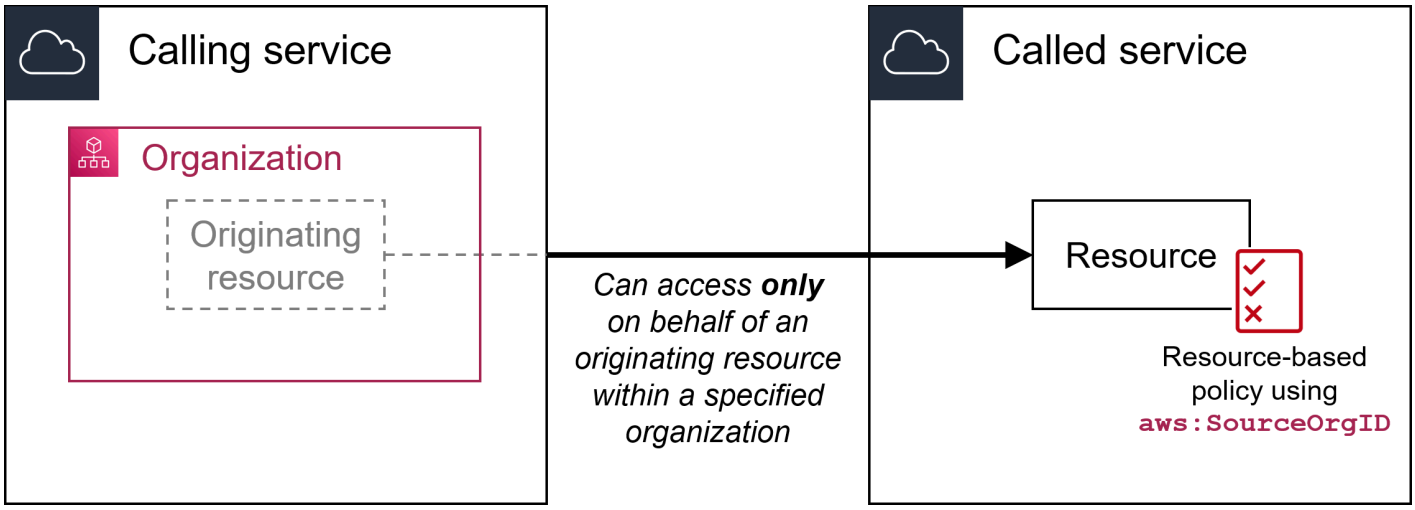
`aws:SourceOrgPaths` adalah kunci kondisi multinilai. Kunci multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan `ForAnyValue` atau `ForAllValues` mengatur operator dengan operator kondisi string untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Kunci konteks multivaluasi.

aws: SourceOrg ID

Gunakan kunci ini untuk membandingkan ID organisasi sumber daya yang membuat service-to-service permintaan dengan ID organisasi yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh prinsipal AWS layanan. Saat Anda menambahkan dan menghapus akun ke organisasi AWS Organizations, kebijakan yang menyertakan `aws:SourceOrgID` kunci secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbarui kebijakan secara manual.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan langsung oleh kepala AWS layanan atas nama sumber daya yang dimiliki

oleh akun yang merupakan anggota organisasi. Layanan panggilan meneruskan ID organisasi dari sumber daya asli ke layanan yang disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan pencatatan akses Elastic Load Balancing di bucket Amazon S3
Semua kepala sekolah layanan	Bot Amazon Lex	Izinkan Layanan AWS untuk menggunakan bot Amazon Lex

> **Note**
>
> Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceOrgID` kebijakan KMS kunci untuk

kunci yang digunakan oleh Layanan AWS melalui hibah KMS kunci dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data - String
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai wakil yang bingung selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan `Principal` prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke ID organisasi sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi SNS topik Amazon, layanan Amazon S3 akan memanggil operasi. `sns:Publish` API Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke ID organisasi bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

aws: UserAgent

Gunakan kunci ini untuk membandingkan aplikasi klien pemohon dan aplikasi yang Anda tentukan dalam kebijakan.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data - String
- Jenis nilai - Bernilai tunggal

⚠ Warning

Kunci ini harus digunakan dengan hati-hati. Karena `aws:UserAgent` nilai diberikan oleh penelepon di HTTP header, pihak yang tidak berwenang dapat menggunakan browser yang dimodifikasi atau kustom untuk memberikan `aws:UserAgent` nilai apa pun yang mereka pilih. Akibatnya, tidak `aws:UserAgent` boleh digunakan untuk mencegah pihak yang tidak berwenang membuat AWS permintaan langsung. Anda dapat menggunakannya hanya untuk mengizinkan aplikasi klien tertentu, dan hanya setelah menguji kebijakan Anda.

Kunci kondisi lintas layanan lainnya

AWS STS mendukung kunci kondisi federasi SAML berbasis dan kunci kondisi lintas layanan untuk OIDCfederasi. Kunci ini tersedia ketika pengguna yang difederasi menggunakan SAML melakukan AWS operasi di layanan lain.

IAMdan kunci konteks AWS STS kondisi

Anda dapat menggunakan `Condition` elemen dalam JSON kebijakan untuk menguji nilai kunci yang disertakan dalam konteks permintaan semua AWS permintaan. Kunci-kunci ini menyediakan informasi tentang permintaannya sendiri atau sumber daya yang dirujuk oleh permintaan. Anda dapat memeriksa bahwa kunci telah menentukan nilai sebelum mengizinkan tindakan yang diminta oleh pengguna. Ini memberi Anda kontrol terperinci atas kapan pernyataan JSON kebijakan Anda cocok atau tidak cocok dengan permintaan yang masuk. Untuk informasi tentang cara menggunakan `Condition` elemen dalam JSON kebijakan, lihatIAMJSONelemen kebijakan: Condition.

Topik ini menjelaskan kunci yang ditentukan dan disediakan oleh IAM layanan (dengan `iam:` awalan) dan layanan AWS Security Token Service (AWS STS) (dengan `sts:` awalan). Beberapa AWS layanan lain juga menyediakan kunci khusus layanan yang relevan dengan tindakan dan sumber daya yang ditentukan oleh layanan tersebut. Untuk informasi selengkapnya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan. Dokumentasi untuk layanan yang mendukung kunci kondisi sering kali memiliki informasi tambahan. Misalnya, untuk informasi tentang kunci yang dapat Anda gunakan dalam kebijakan untuk sumber daya Amazon S3, lihat Kunci Kebijakan Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Topik

- Kunci yang tersedia untuk IAM
- Kunci yang tersedia untuk AWS OIDC federasi
- Kunci yang tersedia untuk AWS STS federasi SAML berbasis
- Kunci konteks AWS STS federasi SAML berbasis lintas layanan
- Kunci yang tersedia untuk AWS STS

Kunci yang tersedia untuk IAM

Anda dapat menggunakan kunci kondisi berikut dalam kebijakan yang mengontrol akses ke IAM sumber daya:

saya: AssociatedResourceArn

Bekerja dengan ARNoperator.

Menentukan ARN sumber daya yang peran ini akan dikaitkan di layanan tujuan. Sumber daya biasanya dimiliki oleh layanan yang perannya diteruskan oleh prinsipal. Terkadang, sumber daya mungkin dimiliki oleh layanan ketiga. Misalnya, Anda dapat meneruskan peran ke Amazon EC2 Auto Scaling yang mereka gunakan pada instans AmazonEC2. Dalam hal ini, kondisinya akan cocok dengan EC2 instance Amazon. ARN

Kunci kondisi ini hanya berlaku untuk PassRoletindakan dalam kebijakan. Kondisi ini tidak dapat digunakan untuk membatasi tindakan lainnya.

Gunakan kunci kondisi ini dalam kebijakan untuk mengizinkan entitas meneruskan peran, tetapi hanya jika peran tersebut terkait dengan sumber daya yang ditentukan. Anda dapat menggunakan wildcard (*) untuk memungkinkan operasi dilakukan pada tipe sumber daya yang spesifik tanpa membatasi Wilayah atau ID sumber daya. Misalnya, Anda dapat mengizinkan IAM pengguna atau peran untuk meneruskan peran apa pun ke EC2 layanan Amazon untuk digunakan dengan instance di Wilayah `us-east-1` atauus-west-1. IAMPengguna atau peran tidak akan diizinkan untuk meneruskan peran ke layanan lain. Selain itu, Amazon tidak mengizinkan Amazon EC2 untuk menggunakan peran dengan instance di Wilayah lain.

```
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {"iam:PassedToService": "ec2.amazonaws.com"},
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:us-east-1:111122223333:instance/*",
                "arn:aws:ec2:us-west-1:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> AWS layanan yang mendukung iam: PassedToService juga mendukung kunci kondisi ini.

saya: AWSServiceName

Bekerja dengan operator string.

Menentukan AWS layanan yang peran ini dilampirkan.

Dalam contoh ini, Anda mengizinkan entitas untuk membuat peran terkait layanan jika nama layanan adalah access-analyzer.amazonaws.com.

```
{
   "Version": "2012-10-17",
   "Statement": [{
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "*",
       "Condition": {
         "StringLike": {
           "iam:AWSServiceName": "access-analyzer.amazonaws.com"
         }
       }
     }]
 }
```

iam: FIDO -sertifikasi

Bekerja dengan operator string.

Memeriksa tingkat FIDO sertifikasi MFA perangkat pada saat pendaftaran kunci FIDO keamanan. Sertifikasi perangkat diambil dari FIDOAlliance Metadata Service (). MDS Jika status sertifikasi atau tingkat kunci FIDO keamanan Anda berubah, itu tidak akan diperbarui kecuali perangkat tidak terdaftar dan terdaftar lagi untuk mengambil informasi sertifikasi yang diperbarui.

Nilai yang mungkin dari L1, L1plus, L2, L2plus, L3, L3plus

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIDO Level 1 plus untuk perangkat Anda.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
    ]

}
```

iam: FIDO - FIPS -140-2-sertifikasi

Bekerja dengan operator string.

Memeriksa MFA perangkat FIPS -140-2 tingkat sertifikasi validasi pada saat pendaftaran kunci keamanan. FIDO Sertifikasi perangkat diambil dari FIDOAlliance Metadata Service (). MDS Jika status sertifikasi atau tingkat kunci FIDO keamanan Anda berubah, itu tidak akan diperbarui kecuali perangkat tidak terdaftar dan terdaftar lagi untuk mengambil informasi sertifikasi yang diperbarui.

Nilai yang mungkin dari L1, L2, L3, L4

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIPS -140-2 Level 2 untuk perangkat Anda.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]

}
```

iam: FIDO - FIPS -140-3-sertifikasi

Bekerja dengan operator string.

Memeriksa MFA perangkat FIPS -140-3 tingkat sertifikasi validasi pada saat pendaftaran kunci keamanan. FIDO Sertifikasi perangkat diambil dari FIDOAlliance Metadata Service (). MDS Jika status sertifikasi atau tingkat kunci FIDO keamanan Anda berubah, itu tidak akan diperbarui kecuali perangkat tidak terdaftar dan terdaftar lagi untuk mengambil informasi sertifikasi yang diperbarui.

Nilai yang mungkin dari L1, L2, L3, L4

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIPS -140-3 Level 3 untuk perangkat Anda.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
                }
            }
        }
    ]

}
```

saya: RegisterSecurityKey

Bekerja dengan operator string.

Memeriksa status pemberdayaan MFA perangkat saat ini.

Nilai yang mungkin dari `Create` atau`Activate`.

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIPS -140-3 Level 1 untuk perangkat Anda.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]

}
```

saya: OrganizationsPolicyId

Bekerja dengan operator string.

Memeriksa apakah kebijakan dengan AWS Organizations ID yang ditentukan cocok dengan kebijakan yang digunakan dalam permintaan. Untuk melihat contoh IAM kebijakan yang menggunakan kunci kondisi ini, lihatIAM: Melihat informasi layanan yang terakhir diakses untuk kebijakan Organizations.

saya: PassedToService

Bekerja dengan operator string.

Menentukan prinsipal layanan untuk peran yang dapat diteruskan. Kunci kondisi ini hanya berlaku untuk PassRoletindakan dalam kebijakan. Kondisi ini tidak dapat digunakan untuk membatasi tindakan lainnya.

Saat Anda menggunakan kunci kondisi ini dalam suatu kebijakan, tentukan layanan menggunakan prinsipal layanan. Prinsipal layanan adalah nama layanan yang dapat ditentukan dalam elemen `Principal` dari suatu kebijakan. Ini adalah format biasa: `SERVICE_NAME_URL.amazonaws.com`.

Anda dapat menggunakan `iam:PassedToService` untuk membatasi pengguna Anda sehingga mereka dapat meneruskan peran ke layanan tertentu. Misalnya, pengguna dapat membuat peran layanan yang dipercaya CloudWatch untuk menulis data log ke bucket Amazon S3

atas nama mereka. Kemudian pengguna harus memberlakukan kebijakan izin dan kebijakan kepercayaan pada peran layanan baru. Dalam hal ini, kebijakan kepercayaan harus menyebutkan `cloudwatch.amazonaws.com` dalam elemen `Principal`. Untuk melihat kebijakan yang memungkinkan pengguna meneruskan peran CloudWatch, lihatIAM: Lulus peran IAM ke layanan tertentu AWS.

Dengan menggunakan kunci kondisi ini, Anda dapat memastikan bahwa pengguna membuat peran layanan hanya untuk layanan yang Anda tentukan. Misalnya, jika pengguna dengan kebijakan sebelumnya mencoba membuat peran layanan untuk AmazonEC2, operasi akan gagal. Kegagalan terjadi karena pengguna tidak memiliki izin untuk meneruskan peran ke AmazonEC2.

Kadang-kadang Anda meneruskan peran ke layanan yang kemudian memberikan peran ke layanan lain. `iam:PassedToService` hanya mencakup layanan akhir yang mengasumsikan peran, bukan layanan perantara yang meneruskan peran.

> **Note**
>
> Beberapa layanan tidak mendukung kunci kondisi ini.

saya: PermissionsBoundary

Bekerja dengan ARNoperator.

Memeriksa apakah kebijakan yang ditentukan dilampirkan sebagai batas izin pada sumber daya utama. IAM Untuk informasi selengkapnya, silakan lihat Batas izin untuk entitas IAM

IAM: Kebijakan ARN

Bekerja dengan ARNoperator.

Memeriksa Amazon Resource Name (ARN) dari kebijakan terkelola dalam permintaan yang melibatkan kebijakan terkelola. Untuk informasi selengkapnya, lihat Mengendalikan akses ke kebijakan.

saya:/ResourceTag***key-name***

Bekerja dengan operator string.

Memeriksa apakah tanda yang dilampirkan ke sumber daya identitas (pengguna atau peran) cocok dengan nama dan nilai kunci yang ditentukan.

> **Note**
>
> IAMdan AWS STS mendukung kunci `iam:ResourceTag` IAM kondisi dan kunci kondisi `aws:ResourceTag` global.

Anda dapat menambahkan atribut kustom ke IAM sumber daya dalam bentuk pasangan kunci-nilai. Untuk informasi selengkapnya tentang tag untuk IAM sumber daya, lihatthe section called "Tag untuk IAM sumber daya". Anda dapat menggunakan `ResourceTag` untuk mengontrol akses ke AWS sumber daya, termasuk IAM sumber daya. Namun, karena IAM tidak mendukung tag untuk grup, Anda tidak dapat menggunakan tag untuk mengontrol akses ke grup.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan menghapus pengguna dengan tag. **`status=terminated`** Untuk menggunakan kebijakan ini, ganti *`italicized placeholder text`* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status": "terminated"}}
    }]
}
```

Kunci yang tersedia untuk AWS OIDC federasi

Anda dapat menggunakan OIDC federasi untuk memberikan kredensyal keamanan sementara kepada pengguna yang telah diautentikasi melalui penyedia identitas kompatibel OpenID Connect (IDP) ke penyedia identitas IAM OpenID Connect () di akun Anda. OIDC AWS Contoh penyedia tersebut termasuk GitHub, Amazon Cognito, Login with Amazon, dan Google. Token identitas dan token akses dari IDP Anda sendiri dapat digunakan, serta token akun layanan yang diberikan kepada beban kerja Amazon Elastic Kubernetes Service.

Anda dapat menggunakan kunci konteks AWS OIDC kondisi untuk menulis kebijakan yang membatasi akses pengguna federasi ke sumber daya yang terkait dengan

penyedia, aplikasi, atau pengguna tertentu. Kunci ini biasanya digunakan dalam kebijakan kepercayaan untuk suatu peran. Tentukan kunci kondisi menggunakan nama OIDC penyedia (`token.actions.githubusercontent.com`) diikuti dengan klaim (`:aud`):**`token.actions.githubusercontent.com:aud`**.

Beberapa kunci kondisi OIDC federasi dapat digunakan dalam sesi peran untuk mengotorisasi akses sumber daya. Jika nilainya adalah Ya di kolom Tersedia dalam sesi, Anda dapat menggunakan kunci kondisi ini dalam kebijakan untuk menentukan pengguna yang diizinkan mengakses AWS layanan lain. Jika klaim tidak tersedia dalam sesi, kunci konteks OIDC kondisi hanya dapat digunakan dalam kebijakan kepercayaan peran untuk AssumeRoleWithWebIdentityotentikasi awal.

Pilih iDP Anda untuk melihat bagaimana klaim dari peta IDP Anda ke IAM mengkondisikan kunci konteks. AWS

Default

Default mencantumkan OIDC klaim standar dan cara mereka memetakan ke AWS STS mengkondisikan kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom JWTklaim iDP. Gunakan pemetaan ini jika IDP Anda tidak tercantum dalam opsi tab.

GitHub Alur kerja tindakan dan Google adalah beberapa contoh IdPs yang menggunakan implementasi default dalam token OIDC JWT ID mereka.

AWS STS kunci kondisi	Klaim JWT IDP	Tersedia dalam sesi
amr	amr	Ya
aud	azp Jika tidak ada nilai yang ditetapkan`azp`, kunci `aud` kondisi akan dipetakan ke `aud` klaim.	Ya
Email	Email	Tidak
oaud	aud	Tidak
sub	sub	Ya

Untuk informasi selengkapnya tentang menggunakan kunci konteks OIDC kondisi dengan GitHub, lihatMengkonfigurasi peran untuk penyedia GitHub OIDC identitas. Untuk informasi selengkapnya tentang kolom `aud` dan `azp` Google, lihat Panduan Google Identity Platform OpenID Connect.

amr

Bekerja dengan operator string. Kunci ini memiliki banyak nilai, artinya jika Anda mengujinya dalam suatu kebijakan yang menggunakan operator kumpulan kondisi.

Contoh: `token.actions.githubusercontent.com:amr`

Referensi Metode Otentikasi mencakup informasi login tentang pengguna. Kunci dapat berisi nilai-nilai berikut:

- Jika pengguna tidak autentikasi, kunci hanya berisi `unauthenticated`.
- Jika pengguna diautentikasi, kunci berisi nilai `authenticated` dan nama penyedia login yang digunakan dalam panggilan (`accounts.google.com`).

aud

Bekerja dengan operator string.

Contoh:

- `accounts.google.com:aud`
- `token.actions.githubusercontent.com:aud`

Gunakan tombol `aud` kondisi untuk memverifikasi bahwa audiens cocok dengan yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci aud dengan sub kunci untuk penyedia identitas yang sama.

Kunci kondisi ini diatur dari bidang token berikut:

- `aud`untuk OAuth 2.0 klien IDs Google aplikasi Anda, ketika `azp` bidang tidak disetel. Ketika `azp` bidang diatur, `aud` bidang cocok dengan kunci `accounts.google.com:oaud` kondisi.
- `azp` jika kolom `azp` ditetapkan. Ini dapat terjadi untuk aplikasi hybrid di mana aplikasi web dan aplikasi Android memiliki ID klien Google OAuth 2.0 yang berbeda tetapi berbagi APIs proyek Google yang sama.

Ketika Anda menulis kebijakan menggunakan kunci kondisi `accounts.google.com:aud`, Anda harus mengetahui apakah aplikasi tersebut merupakan aplikasi hibrida yang menetapkan kolom `azp` tersebut atau bukan.

azpKolom Tidak Ditetapkan

Contoh kebijakan berikut bekerja untuk aplikasi non-hibrida yang tidak menetapkan kolom azp. Dalam hal ini, nilai kolom Token ID Google aud sesuai dengan nilai kunci kondisi accounts.google.com:aud dan accounts.google.com:oaud.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

Kumpulan Kolom azp

Contoh kebijakan berikut berfungsi untuk aplikasi non-hibrida yang tidak menetapkan kolom azp. Dalam hal ini, nilai kolom aud Token ID Google hanya sesuai dengan nilai kunci kondisi accounts.google.com:oaud. Nilai kolom azp sesuai dengan nilai kunci kondisi accounts.google.com:aud.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
```

```
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

email

Bekerja dengan operator string.

Contoh: `accounts.google.com:email`

Kunci kondisi ini memvalidasi alamat email pengguna. Nilai klaim ini mungkin tidak unik untuk akun ini dan dapat berubah seiring waktu, oleh karena itu Anda tidak boleh menggunakan nilai ini sebagai pengenal utama untuk memverifikasi catatan pengguna Anda.

oaud

Bekerja dengan operator string.

Contoh: `accounts.google.com:oaud`

Kunci ini menentukan audience (`aud`) lain yang dimaksudkan untuk token ID ini. Itu harus menjadi salah satu klien OAuth 2.0 IDs dari aplikasi Anda.

sub

Bekerja dengan operator string.

Contoh:

- `accounts.google.com:sub`
- token.actions.githubusercontent.com:sub

Gunakan kunci ini untuk memverifikasi bahwa subjek cocok dengan yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci `sub` dengan kunci `aud` untuk penyedia identitas yang sama.

Dalam kebijakan kepercayaan peran berikut, kunci `sub` kondisi membatasi peran ke GitHib cabang bernama`demo`.

```
{
   "Version": "2012-10-17",
   "Statement": [
```

```
      "Condition": {
         "StringEquals": {
            "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
            "token.actions.githubusercontent.com:sub": "repo:octo-org/octo-
repo:ref:refs/heads/demo"
         }
      }
   ]
}
```

Amazon Cognito

Tab ini menjelaskan bagaimana Amazon Cognito memetakan OIDC klaim untuk AWS STS mengkondisikan kunci konteks di. AWS Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom JWTklaim iDP.

Untuk peran yang digunakan oleh Amazon Cognito, kunci ditentukan menggunakan `cognito-identity.amazonaws.com` diikuti oleh klaim.

Untuk informasi selengkapnya tentang pemetaan klaim kumpulan identitas, lihat Pemetaan penyedia default di Panduan Pengembang Amazon Cognito. Untuk informasi selengkapnya tentang pemetaan klaim kumpulan pengguna, lihat Menggunakan token ID di Panduan Pengembang Amazon Cognito.

AWS STS kunci kondisi	Klaim JWT IDP	Tersedia dalam sesi
amr	amr	Ya
aud	aud	Ya
oaud	aud	Tidak
sub	sub	Ya

amr

Bekerja dengan operator string. Kunci ini memiliki banyak nilai, artinya jika Anda mengujinya dalam suatu kebijakan yang menggunakan operator kumpulan kondisi.

Contoh - `cognito-identity.amazonaws.com:amr`

Referensi Metode Otentikasi mencakup informasi login tentang pengguna. Kunci dapat berisi nilai-nilai berikut:

- Jika pengguna tidak autentikasi, kunci hanya berisi `unauthenticated`.
- Jika pengguna diautentikasi, kunci berisi nilai `authenticated` dan nama penyedia login yang digunakan dalam panggilan (`cognito-identity.amazonaws.com`).

Sebagai contoh, kondisi berikut dalam kebijakan kepercayaan untuk peran Amazon Cognito menguji apakah pengguna tidak diautentikasi.

```
"Condition": {
  "StringEquals":
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

aud

Bekerja dengan operator string.

Contoh - `cognito-identity.amazonaws.com:aud`

Klien aplikasi kumpulan pengguna yang mengautentikasi pengguna Anda. Amazon Cognito memberikan nilai yang sama dalam klaim token akses. `client_id`

oaud

Bekerja dengan operator string.

Contoh - `cognito-identity.amazonaws.com:oaud`

Klien aplikasi kumpulan pengguna yang mengautentikasi pengguna Anda. Amazon Cognito memberikan nilai yang sama dalam klaim token akses. `client_id`

sub

Bekerja dengan operator string.

Contoh - `cognito-identity.amazonaws.com:sub`

Pengenal unik (UUID), atau subjek, untuk pengguna yang diautentikasi. Nama pengguna mungkin tidak unik di kumpulan pengguna Anda. Sub klaim adalah cara terbaik untuk

mengidentifikasi pengguna tertentu. Anda dapat menggunakan kunci sub dengan kunci aud untuk penyedia identitas yang sama.

```
{
   "Version": "2012-10-17",
   "Statement": [
      "Condition": {
         "StringEquals": {
            "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
            "cognito-identity.amazonaws.com:sub": [
               "us-east-1:12345678-1234-1234-1234-123456790ab",
               "us-east-1:98765432-1234-1234-1243-123456790ab"
            ]
         }
      }
   ]
}
```

Login with Amazon

Tab ini menjelaskan bagaimana Login with Amazon maps OIDC mengklaim AWS STS mengkondisikan kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom JWTklaim iDP.

AWS STS kunci kondisi	Klaim JWT IDP	Tersedia dalam sesi
app_id	ID Aplikasi	Ya
sub	ID Pengguna	Ya
user_id	ID Pengguna	Ya

app_id

Bekerja dengan operator string.

Contoh - `www.amazon.com:app_id`

Kunci ini menentukan konteks audiens yang cocok dengan `aud` bidang yang digunakan oleh penyedia identitas lainnya.

sub

Bekerja dengan operator string.

Contoh - `www.amazon.com:sub`

Kunci ini memverifikasi bahwa ID pengguna cocok dengan yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci `sub` dengan kunci `aud` untuk penyedia identitas yang sama.

user_id

Bekerja dengan operator string.

Contoh - `www.amazon.com:user_id`

Kunci ini menentukan konteks audiens yang cocok dengan `aud` bidang yang digunakan oleh penyedia identitas lain.Anda dapat menggunakan `user_id` kunci dengan `id` kunci untuk penyedia identitas yang sama.

Facebook

Tab ini menjelaskan bagaimana Facebook memetakan OIDC klaim untuk AWS STS mengkondisikan kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom JWTklaim iDP.

AWS STS kunci kondisi	Klaim JWT IDP	Tersedia dalam sesi
app_id	ID Aplikasi	Ya
id	id	Ya

app_id

Bekerja dengan operator string.

Contoh - `graph.facebook.com:app_id`

Kunci ini memverifikasi bahwa konteks audiens cocok dengan `aud` bidang yang digunakan oleh penyedia identitas lain.

id

Bekerja dengan operator string.

Contoh - `graph.facebook.com:id`

Kunci ini memverifikasi bahwa ID aplikasi (atau situs) cocok dengan yang Anda tentukan dalam kebijakan.

Informasi lebih lanjut tentang OIDC federasi

- Panduan Pengguna Amazon Cognito
- OIDCfederasi

Kunci yang tersedia untuk AWS STS federasi SAML berbasis

Jika Anda bekerja dengan federasi SAML berbasis menggunakan AWS Security Token Service (AWS STS), Anda dapat menyertakan kunci kondisi tambahan dalam kebijakan.

SAMLkebijakan kepercayaan peran

Dalam kebijakan kepercayaan dari suatu peran, Anda dapat menyertakan kunci-kunci berikut yang membantu Anda menentukan apakah pemanggil diperbolehkan untuk memegang peran tersebut. Kecuali`saml:doc`, semua nilai berasal dari SAML pernyataan. Semua item dalam daftar tersedia di editor visual IAM konsol saat Anda membuat atau mengedit kebijakan dengan kondisi. Item yang ditandai dengan `[]` dapat memiliki nilai yang merupakan daftar tipe yang ditentukan.

sml:aud

Bekerja dengan operator string.

Titik akhir URL di mana SAML pernyataan disajikan. Nilai untuk kunci ini berasal dari kolom `SAML Recipient` dalam pernyataan, bukan kolom `Audience`.

sama: commonName []

Bekerja dengan operator string.

Ini adalah atribut `commonName`.

saml:cn[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:doc

Bekerja dengan operator string.

Ini mewakili prinsipal yang digunakan untuk menerima peran tersebut. Formatnya adalah *`account-ID`*/*`provider-friendly-name`*, seperti`123456789012/SAMLProviderName`. Nilai Account-ID mengacu pada akun yang memiliki penyedia. SAML

saml:edupersonaffiliation[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonassurance[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonentitlement[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonnickname[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonorgdn

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonorgunitdn[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonprimaryaffiliation

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonprimaryorgunitdn

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonprincipalname

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonscopedaffiliation[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersontargetedid[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:eduorghomepageuri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorgidentityauthnpolicyuri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorglegalname[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorgsuperioruri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorgwhitepagesuri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

sama: givenName []

Bekerja dengan operator string.

Ini adalah atribut `givenName`.

sml:iss

Bekerja dengan operator string.

Penerbit, yang diwakili oleh aURN.

saml:mail[]

Bekerja dengan operator string.

Ini adalah atribut `mail`.

saml:name[]

Bekerja dengan operator string.

Ini adalah atribut `name`.

saml:namequalifier

Bekerja dengan operator string.

Nilai hash berdasarkan nama ramah SAML penyedia. Nilai adalah rangkaian dari nilai-nilai berikut, diurutkan dan dipisahkan oleh karakter '/':

1. Nilai tanggapan `Issuer` (`saml:iss`)
2. ID akun `AWS`
3. Nama ramah (bagian terakhir dariARN) SAML penyedia di IAM

Gabungan ID akun dan nama ramah SAML penyedia tersedia untuk IAM kebijakan sebagai kuncinya. `saml:doc` Untuk informasi selengkapnya, lihat Mengidentifikasi pengguna secara unik di SAML federasi berbasis.

sama: organizationStatus []

Bekerja dengan operator string.

Ini adalah atribut `organizationStatus`.

sama: primaryGroup SID []

Bekerja dengan operator string.

Ini adalah atribut `primaryGroupSID`.

saml:sub

Bekerja dengan operator string.

Ini adalah subjek klaim, yang mencakup nilai yang secara unik mengidentifikasi pengguna individu dalam suatu organisasi (misalnya, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Bekerja dengan operator string.

Kunci ini dapat memiliki nilai`persistent`,`transient`, atau terdiri dari penuh `Format` URI dari `NameID` elemen `Subject` dan yang digunakan dalam SAML pernyataan Anda. Nilai dari `persistent` menunjukkan bahwa nilai dalam `saml:sub` sama untuk pengguna di antara sesi. Jika nilainya `transient`, pengguna memiliki nilai `saml:sub` untuk setiap sesi. Untuk informasi tentang elemen `NameID` `Format` atribut, lihat Konfigurasikan SAML pernyataan untuk respons otentikasi.

saml:surname[]

Bekerja dengan operator string.

Ini adalah atribut `surnameuid`.

saml:uid[]

Bekerja dengan operator string.

Ini adalah atribut `uid`.

saml:x500 [] UniqueIdentifier

Bekerja dengan operator string.

Ini adalah atribut `x500UniqueIdentifier`.

Untuk informasi umum tentang `eduPerson` dan `eduOrg` atribut, lihat situs web REFEDS Wiki. Untuk daftar `eduPerson` atribut, lihat Spesifikasi Kelas eduPerson Objek (201602).

Kunci kondisi dengan tipe berupa daftar dapat mencakup beberapa nilai. Untuk membuat kondisi dalam kebijakan untuk nilai-nilai daftar, Anda dapat menggunakan operator kumpulan (`ForAllValues`, `ForAnyValue`). Misalnya, untuk mengizinkan pengguna yang berafiliasi dengan "fakultas" atau "staf" (tetapi bukan "mahasiswa"), Anda dapat menggunakan ketentuan seperti berikut:

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

Kunci konteks AWS STS federasi SAML berbasis lintas layanan

Beberapa kunci kondisi federasi SAML berbasis dapat digunakan dalam permintaan berikutnya untuk mengotorisasi AWS operasi di layanan dan `AssumeRole` panggilan lain. Ini adalah kunci kondisi berikut yang dapat digunakan dalam kebijakan kepercayaan peran ketika kepala sekolah federasi mengambil peran lain, dan dalam kebijakan sumber daya dari AWS layanan lain untuk mengotorisasi akses sumber daya oleh prinsipal federasi. Untuk informasi selengkapnya tentang penggunaan kunci ini, lihat Tentang federasi SAML berbasis 2.0.

Pilih tombol kondisi untuk melihat deskripsi.

- saml:namequalifier

- saml:sub
- saml:sub_type

> **Note**
>
> Tidak ada kunci kondisi federasi SAML berbasis lain yang tersedia untuk digunakan setelah respons otentikasi penyedia identitas eksternal (iDP) awal.

Kunci yang tersedia untuk AWS STS

Anda dapat menggunakan kunci kondisi berikut dalam kebijakan kepercayaan IAM peran untuk peran yang diasumsikan menggunakan operasi AWS Security Token Service (AWS STS).

saml:sub

Bekerja dengan operator string.

Ini adalah subjek klaim, yang mencakup nilai yang secara unik mengidentifikasi pengguna individu dalam suatu organisasi (misalnya, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

sts: AWSServiceName

Bekerja dengan operator string.

Gunakan kunci ini untuk menentukan layanan di mana token pembawa dapat digunakan. Saat Anda menggunakan kunci kondisi ini dalam suatu kebijakan, tentukan layanan menggunakan prinsipal layanan. Prinsipal layanan adalah nama layanan yang dapat ditentukan dalam elemen `Principal` dari suatu kebijakan. Misalnya, `codeartifact.amazonaws.com` adalah kepala AWS CodeArtifact layanan.

Ketersediaan – Kunci ini tersedia dalam permintaan yang mendapatkan token pembawa. Anda tidak dapat melakukan panggilan langsung AWS STS untuk mendapatkan token pembawa. Saat Anda melakukan beberapa operasi dalam layanan lain, layanan meminta token pembawa atas nama Anda.

Beberapa AWS layanan mengharuskan Anda memiliki izin untuk mendapatkan token pembawa AWS STS layanan sebelum Anda dapat mengakses sumber daya mereka secara terprogram. Misalnya, AWS CodeArtifact mengharuskan prinsipal untuk menggunakan token pembawa untuk

melakukan beberapa operasi. Perintah `aws codeartifact get-authorization-token` mengembalikan token pembawa. Anda kemudian dapat menggunakan token pembawa untuk melakukan AWS CodeArtifact operasi. Untuk informasi selengkapnya tentang token pembawa, lihat Token pembawa layanan.

Anda dapat menggunakan kunci kondisi ini untuk mengizinkan prinsipal mendapatkan token pembawa untuk digunakan dengan layanan tertentu.

sts: DurationSeconds

Bekerja dengan operator numerik.

Gunakan kunci ini untuk menentukan durasi (dalam detik) yang dapat digunakan prinsipal saat mendapatkan token AWS STS pembawa.

Ketersediaan – Kunci ini tersedia dalam permintaan yang mendapatkan token pembawa. Anda tidak dapat melakukan panggilan langsung AWS STS untuk mendapatkan token pembawa. Saat Anda melakukan beberapa operasi dalam layanan lain, layanan meminta token pembawa atas nama Anda. Kuncinya tidak tersedia untuk operasi assume-role AWS STS .

Beberapa AWS layanan mengharuskan Anda memiliki izin untuk mendapatkan token pembawa AWS STS layanan sebelum Anda dapat mengakses sumber daya mereka secara terprogram. Misalnya, AWS CodeArtifact mengharuskan prinsipal untuk menggunakan token pembawa untuk melakukan beberapa operasi. Perintah `aws codeartifact get-authorization-token` mengembalikan token pembawa. Anda kemudian dapat menggunakan token pembawa untuk melakukan AWS CodeArtifact operasi. Untuk informasi selengkapnya tentang token pembawa, lihat Token pembawa layanan.

sts: ExternalId

Bekerja dengan operator string.

Gunakan kunci ini untuk mengharuskan prinsipal memberikan pengenal tertentu saat mengambil peranIAM.

Ketersediaan — Kunci ini hadir dalam permintaan ketika prinsipal memberikan ID eksternal sambil mengambil peran menggunakan AWS CLI or AWS API.

Pengidentifikasi unik yang mungkin diperlukan saat Anda menerima peran dalam akun lain. Jika administrator akun yang memiliki peran tersebut memberi Anda ID eksternal, berikan nilai tersebut di parameter `ExternalId`. Nilai ini dapat berupa string, seperti frasa sandi atau nomor

akun. Fungsi utama dari ID eksternal adalah mengatasi dan mencegah masalah deputi yang membingungkan. Untuk informasi selengkapnya tentang ID eksternal dan masalah deputi yang membingungkan, lihat Akses ke Akun AWS yang dimiliki oleh pihak ketiga.

Nilai `ExternalId` harus memiliki minimal 2 karakter dan maksimal 1.224 karakter. Nilai harus berupa alfanumerik tanpa spasi. Alfanumerik dapat mencakup simbol berikut: plus (+), sama (=), koma (,), titik (.), di (@), titik dua (:), garis miring (/), dan tanda hubung (-).

sts:RequestContext/kunci-konteks

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan pasangan nilai kunci konteks sesi yang disematkan dalam pernyataan konteks yang ditandatangani penerbit token tepercaya yang diteruskan dalam permintaan dengan nilai kunci konteks yang ditentukan dalam kebijakan kepercayaan peran.

Ketersediaan — Kunci ini hadir dalam permintaan ketika pernyataan konteks disediakan dalam parameter `ProvidedContexts` permintaan sambil mengasumsikan peran menggunakan operasi. AWS STS AssumeRoleAPI

Kunci konteks ini diformat sebagai `"sts:RequestContext/context-key":"context-value"` where `context-key` dan `context-value` merupakan pasangan kunci-nilai konteks. Ketika beberapa kunci konteks disematkan dalam pernyataan konteks yang ditandatangani yang diteruskan dalam permintaan, ada satu kunci konteks untuk setiap pasangan kunci-nilai. Anda harus memberikan izin untuk `sts:SetContext` tindakan dalam kebijakan kepercayaan peran untuk mengizinkan prinsipal menyetel kunci konteks dalam token sesi yang dihasilkan. Untuk mempelajari selengkapnya tentang kunci konteks Pusat IAM Identitas yang didukung yang dapat digunakan dengan kunci ini, lihat tombol AWS STS kondisi untuk Pusat IAM Identitas di Panduan AWS IAM Identity Center Pengguna.

Anda dapat menggunakan kunci ini dalam kebijakan kepercayaan peran untuk menerapkan kontrol akses berbutir halus berdasarkan pengguna atau atributnya saat mereka mengambil peran. Setelah peran diasumsikan, aktivitas akan muncul di AWS CloudTrail log dalam `AdditionalEventData` atribut, yang berisi pasangan nilai kunci konteks sesi yang ditetapkan oleh penyedia konteks dalam permintaan peran asumsikan. Ini memudahkan administrator membedakan di antara sesi peran saat peran digunakan oleh prinsipal yang berbeda. Pasangan kunci-nilai ditetapkan oleh penyedia konteks yang ditentukan, bukan oleh AWS CloudTrail atau. AWS STS Ini memberi penyedia konteks kontrol atas konteks apa yang disertakan dalam CloudTrail log dan informasi sesi.

sts: RequestContextProviders

Bekerja dengan ARNoperator.

Gunakan kunci ini untuk membandingkan penyedia konteks ARN dalam permintaan dengan penyedia konteks ARN yang ditentukan dalam kebijakan kepercayaan peran.

Ketersediaan — Kunci ini hadir dalam permintaan ketika pernyataan konteks disediakan dalam parameter `ProvidedContexts` permintaan sambil mengasumsikan peran menggunakan operasi. AWS STS AssumeRoleAPI

Contoh kondisi berikut memeriksa apakah penyedia konteks yang ARN diteruskan dalam permintaan cocok dengan yang ARN ditentukan dalam kondisi kebijakan kepercayaan peran.

```
"Condition": {
  "ForAllValues:ArnEquals": {
    "sts:RequestContextProviders": [
      "arn:aws:iam::aws:contextProvider/IdentityCenter"
    ]
  }
}
```

sts: RoleSessionName

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan nama sesi yang ditentukan oleh prinsipal saat mengambil peran dengan nilai yang ditentukan dalam kebijakan.

Ketersediaan — Kunci ini hadir dalam permintaan ketika prinsipal mengasumsikan peran menggunakan, CLI perintah peran asumsi apa pun AWS Management Console, atau operasi apa pun. AWS STS `AssumeRole` API

Anda dapat menggunakan kunci ini dalam kebijakan kepercayaan peran untuk mewajibkan pengguna Anda memberikan nama sesi tertentu saat mereka mengambil peran. Misalnya, Anda dapat meminta IAM pengguna menentukan nama pengguna mereka sendiri sebagai nama sesi mereka. Setelah IAM pengguna mengambil peran, aktivitas muncul di AWS CloudTrail log dengan nama sesi yang cocok dengan nama pengguna mereka. Ini memudahkan administrator membedakan di antara sesi peran saat peran digunakan oleh prinsipal yang berbeda.

Kebijakan kepercayaan peran berikut mengharuskan IAM pengguna di akun `111122223333` memberikan nama IAM pengguna mereka sebagai nama sesi saat mereka mengambil peran.

Persyaratan ini diberlakukan menggunakan `aws:username` variabel kondisi di dalam kunci kondisi. Kebijakan ini memungkinkan IAM pengguna untuk mengambil peran yang melekat pada kebijakan tersebut. Kebijakan ini tidak mengizinkan siapa pun yang menggunakan kredensyal sementara untuk mengambil peran karena `username` variabel hanya IAM ada untuk pengguna.

> **Important**
>
> Anda dapat menggunakan kunci kondisi bernilai tunggal sebagai variabel. Anda tidak dapat menggunakan kunci kondisi multivalued sebagai variabel.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "${aws:username}"}
            }
        }
    ]
}
```

Saat administrator melihat AWS CloudTrail log untuk tindakan, mereka dapat membandingkan nama sesi dengan nama pengguna di akun mereka. Dalam contoh berikut, pengguna bernama `matjac` melakukan operasi menggunakan peran bernama `MateoRole`. Administrator selanjutnya dapat menghubungi Mateo Jackson, yang memiliki pengguna bernama `matjac`.

```
    "assumedRoleUser": {
        "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
        "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
    }
```

Jika Anda mengizinkan akses lintas akun menggunakan peran, maka pengguna di satu akun dapat mengambil peran di akun lain. Pengguna ARN peran yang diasumsikan terdaftar CloudTrail termasuk akun tempat peran itu ada. Ini tidak termasuk akun pengguna yang mengambil peran

tersebut. Pengguna hanya bersifat unik di dalam suatu akun. Oleh karena itu, kami menyarankan Anda menggunakan metode ini untuk memeriksa CloudTrail log hanya untuk peran yang diasumsikan oleh pengguna di akun yang Anda kelola. Pengguna Anda mungkin menggunakan nama pengguna yang sama dalam beberapa akun.

sts: SourceIdentity

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan identitas sumber yang ditentukan oleh prinsipal saat mengambil peran dengan nilai yang ditentukan dalam kebijakan.

Ketersediaan — Kunci ini hadir dalam permintaan ketika prinsipal memberikan identitas sumber sambil mengasumsikan peran menggunakan perintah, atau AWS STS operasi peran CLI apa pun. AWS STS `AssumeRole` API

Anda dapat menggunakan kunci ini dalam kebijakan kepercayaan peran untuk mewajibkan pengguna Anda menetapkan identitas sumber tertentu saat mereka mengambil peran. Misalnya, Anda dapat meminta tenaga kerja Anda atau identitas gabungan untuk menentukan nilai identitas sumber. Anda dapat mengonfigurasi penyedia identitas Anda (IdP) untuk menggunakan salah satu atribut yang terkait dengan pengguna Anda, seperti nama pengguna atau email sebagai identitas sumber. IdP kemudian meneruskan identitas sumber sebagai atribut dalam pernyataan atau klaim yang dikirimkan ke. AWS Nilai atribut identitas sumber mengidentifikasi pengguna atau aplikasi yang mengambil peran.

Setelah pengguna mengambil peran, aktivitas muncul di log AWS CloudTrail dengan nilai identitas sumber yang ditetapkan. Ini memudahkan administrator untuk menentukan siapa atau apa yang melakukan tindakan dengan peran AWS. Anda harus memberikan izin untuk tindakan `sts:SetSourceIdentity` untuk mengizinkan identitas menetapkan identitas sumber.

Tidak seperti `sts:RoleSessionName`, setelah identitas sumber diatur, nilai tidak dapat diubah. Ini terdapat dalam konteks permintaan untuk semua tindakan yang diambil dengan peran menggunakan identitas sumber daya. Nilai tetap ada dalam sesi peran berikutnya saat Anda menggunakan kredensial sesi untuk mengambil peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

Anda dapat menggunakan kunci kondisi `aws:SourceIdentity`global untuk mengontrol akses lebih lanjut ke AWS sumber daya berdasarkan nilai identitas sumber dalam permintaan berikutnya.

Kebijakan kepercayaan peran berikut memungkinkan IAM pengguna `AdminUser` untuk mengambil peran dalam akun`111122223333`. Ini juga memberikan izin ke `AdminUser` untuk mengatur identitas sumber, selama identitas sumber yang ditetapkan adalah `DiegoRamirez`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
}
```

Untuk informasi selengkapnya tentang menggunakan informasi identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

sts: TransitiveTagKeys

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan kunci tag sesi transitif dalam permintaan dengan yang ditentukan dalam kebijakan.

Ketersediaan – Kunci ini tersedia dalam permintaan saat Anda mengajukan permintaan menggunakan kredensial keamanan sementara. Ini termasuk kredensial yang dibuat menggunakan operasi asumsi peran, atau operasi `GetFederationToken`.

Saat Anda menggunakan kredensial sesi untuk membuat permintaan berikutnya, konteks permintaan mencakup kunci konteks `aws:PrincipalTag`. Kunci ini mencakup daftar tanda sesi, tanda sesi transitif, dan tanda peran. Tanda sesi transitif adalah tanda yang tetap ada di semua sesi berikutnya saat Anda menggunakan kredensial sesi untuk menjalankan peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

Anda dapat menggunakan kunci kondisi ini dalam kebijakan untuk mewajibkan pengaturan tanda sesi tertentu sebagai transitif saat mengambil peran atau menggabungkan pengguna.

Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan

Setiap AWS layanan dapat menentukan tindakan, sumber daya, dan kunci konteks kondisi untuk digunakan dalam kebijakan IAM. Untuk daftar AWS layanan dan tindakan, sumber daya, dan kunci konteks kondisinya, lihat Tindakan, sumber daya, dan kunci kondisi di Referensi Otorisasi Layanan.

Sumber daya tentang IAM yang dapat dipelajari lebih lanjut

IAM adalah produk yang kaya, dan Anda akan menemukan banyak sumber daya untuk membantu Anda mempelajari lebih lanjut tentang bagaimana IAM dapat membantu Anda mengamankan sumber daya Akun AWS dan sumber daya Anda.

Topik

Identitas

Konsultasikan sumber daya ini untuk membuat, mengelola, dan menggunakan identitas.

- Kelola identitas di Pusat Identitas IAM - Informasi prosedural tentang membuat pengguna dan grup di Pusat Identitas IAM.
- IAMIdentitas — Diskusi mendalam tentang pengguna, grup, dan peran.

Kredensial (RDS, access key, dan perangkat MFA)

Tinjau panduan berikut untuk mengelola kata sandi, kunci akses, dan perangkat MFA untuk Anda Akun AWS dan untuk pengguna IAM.

- Kelola kata sandi pengguna di AWS— Menjelaskan opsi untuk mengelola kata sandi untuk pengguna IAM di akun Anda.
- Mengelola kunci akses untuk IAM pengguna— Menjelaskan cara kerja kunci akses dan bagaimana Anda dapat menggunakannya untuk melakukan panggilan terprogram. AWS Ada alternatif lain yang lebih aman untuk mengakses kunci yang kami sarankan Anda pertimbangkan terlebih dahulu.

Untuk informasi selengkapnya, lihat Pertimbangan dan alternatif untuk kunci akses jangka panjang dalam Referensi Umum AWS panduan ini.

- AWS Otentikasi multi-faktor di IAM – Menjelaskan cara mengonfigurasi akun Anda dan pengguna IAM untuk meminta kata sandi dan kode penggunaan sekali pakai yang dibuat di perangkat sebelum diizinkan masuk. (Ini terkadang disebut sebagai autentikasi dua faktor.)

Untuk informasi umum tentang jenis kredensil yang Anda gunakan untuk mengakses Amazon Web Services, lihat AWS Security Credentials di panduan.Referensi Umum AWS

Izin dan kebijakan

Pelajari cara kerja kebijakan IAM dan temukan tip tentang cara terbaik untuk memberikan izin:

- Kebijakan dan izin di AWS Identity and Access Management – Memperkenalkan bahasa kebijakan yang digunakan untuk mendefinisikan izin. Menjelaskan bagaimana izin dapat dilampirkan ke pengguna atau grup atau, untuk beberapa produk AWS , ke sumber daya itu sendiri.
- IAMJSONreferensi elemen kebijakan – Memberikan deskripsi dan contoh setiap elemen bahasa kebijakan.
- Validasi kebijakan IAM— Temukan sumber daya untuk validasi kebijakan JSON.
- Contoh kebijakan berbasis identitas IAM— Menunjukkan contoh kebijakan untuk tugas umum di berbagai AWS produk.
- AWS Policy Generator - Buat kebijakan khusus dengan memilih produk dan tindakan dari daftar.
- IAM Policy Simulator — Uji apakah suatu kebijakan akan mengizinkan atau menolak permintaan tertentu. AWS

Federasi dan delegasi

Anda dapat memberikan akses ke sumber daya Akun AWS untuk pengguna yang diautentikasi (masuk) di tempat lain. Ini dapat berupa pengguna IAM di pengguna lain Akun AWS (dikenal sebagai delegasi), pengguna yang diautentikasi dengan proses masuk organisasi Anda, atau pengguna dari penyedia identitas Internet seperti Login with Amazon, Facebook, Google, atau penyedia identitas kompatibel OpenID Connect (OIDC) lainnya. Dalam kasus ini, pengguna mendapatkan kredensil keamanan sementara untuk mengakses AWS sumber daya.

- IAMtutorial: Delegasikan akses di seluruh AWS akun menggunakan IAM peran— Memandu Anda melalui pemberian akses lintas akun ke pengguna IAM di pengguna lain. Akun AWS
- Skenario umum untuk kredensial sementara— Menjelaskan cara-cara di mana pengguna dapat digabungkan ke dalam AWS setelah diautentikasi di luar. AWS

IAM dan produk lainnya AWS

Sebagian besar AWS produk terintegrasi dengan IAM sehingga Anda dapat menggunakan fitur IAM untuk membantu melindungi akses ke sumber daya dalam produk tersebut. Sumber daya berikut membahas IAM dan keamanan untuk beberapa AWS produk paling populer. Untuk daftar lengkap produk yang bekerja dengan IAM, termasuk tautan ke informasi lebih lanjut tentang masing-masing, lihat AWS layanan yang bekerja dengan IAM.

Menggunakan IAM dengan Amazon EC2

- Mengendalikan Akses ke Sumber Daya Amazon EC2 – Menjelaskan cara menggunakan fitur IAM untuk mengizinkan pengguna mengelola instans Amazon EC2, volume, dan lainnya.
- Gunakan profil contoh— Menjelaskan cara menggunakan peran IAM untuk menyediakan kredensil secara aman untuk aplikasi yang berjalan di instans Amazon EC2 dan yang memerlukan akses ke produk lain. AWS

Menggunakan IAM dengan Amazon S3

- Mengelola Izin Akses ke Sumber Daya Amazon S3 Anda – Membahas model kerahasiaan Amazon S3 untuk bucket dan objek yang mencakup kebijakan IAM.
- Menulis Kebijakan IAM: Berikan Akses ke Folder Khusus Pengguna di Amazon S3 Bucket — Membahas cara membiarkan pengguna melindungi folder mereka sendiri di Amazon S3. (Untuk posting lebih lanjut tentang Amazon S3 dan IAM, pilih tag S3 di bawah judul postingan blog.)

Menggunakan IAM dengan Amazon RDS

- Menggunakan AWS Identity and Access Management (IAM) untuk Mengelola Akses ke Sumber Daya Amazon RDS — Menjelaskan cara menggunakan IAM untuk mengontrol akses ke instans database, snapshot database, dan lainnya.

- Pelatihan Dasar tentang Izin Tingkat Sumber Daya RDS – Menjelaskan cara menggunakan IAM untuk mengontrol akses ke instans Amazon RDS tertentu.

Menggunakan IAM dengan Amazon DynamoDB

- Menggunakan IAM untuk Mengontrol Akses ke Sumber Daya DynamoDB - Menjelaskan cara menggunakan IAM untuk mengizinkan pengguna mengelola tabel dan indeks DynamoDB.
- Video berikut (8:55) menjelaskan cara menyediakan kontrol akses untuk masing-masing item database DynamoDB atau atribut (atau keduanya).

  Memulai dengan Kontrol Akses Berbutir Halus untuk DynamoDB

Praktik kerahasiaan umum

Temukan kiat dan panduan ahli tentang cara terbaik untuk mengamankan sumber daya Akun AWS dan Anda:

- Praktik Terbaik untuk Keamanan, Identitas, & Kepatuhan — Temukan sumber daya untuk mengelola keamanan Akun AWS dan produk, termasuk saran untuk arsitektur keamanan, penggunaan IAM, enkripsi dan keamanan data, dan banyak lagi.
- Identity and Access Management — The AWS Well-Architected Framework membantu Anda memahami konsep kunci, prinsip desain, dan praktik terbaik arsitektur untuk merancang dan menjalankan beban kerja di cloud.
- Praktik terbaik keamanan di IAM— Menawarkan rekomendasi untuk cara menggunakan IAM untuk membantu mengamankan sumber daya Akun AWS dan Anda.
- AWS CloudTrail Panduan Pengguna — Gunakan AWS CloudTrail untuk melacak riwayat panggilan API yang dilakukan AWS dan menyimpan informasi tersebut dalam file log. Ini membantu Anda menentukan pengguna dan akun mana yang mengakses sumber daya di akun Anda, kapan panggilan dilakukan, tindakan apa yang diminta, dan banyak lagi.

Sumber daya umum

Jelajahi sumber daya berikut untuk mempelajari lebih lanjut tentang IAM dan AWS.

- Informasi Produk untuk IAM — Informasi umum tentang AWS Identity and Access Management produk.

- AWS re:Post untuk AWS Identity and Access Management — Kunjungi AWS re:Post untuk mendiskusikan pertanyaan teknis terkait IAM dengan AWS masyarakat.

- Kelas & Lokakarya - Tautan ke kursus berbasis peran dan khusus, selain laboratorium mandiri untuk membantu mempertajam keterampilan Anda AWS dan mendapatkan pengalaman praktis.
- AWS Pusat Pengembang — Jelajahi tutorial, unduh alat, dan pelajari tentang acara AWS pengembang.
- AWS Alat Pengembang - Tautan ke alat pengembang, SDK, toolkit IDE, dan alat baris perintah untuk mengembangkan dan mengelola aplikasi. AWS
- Memulai Pusat Sumber Daya — Pelajari cara menyiapkan Akun AWS, bergabung dengan AWS komunitas, dan meluncurkan aplikasi pertama Anda.
- Hands-On Tutorial - Ikuti step-by-step tutorial untuk meluncurkan aplikasi pertama Anda. AWS
- AWS Whitepaper — Tautan ke daftar lengkap AWS whitepaper teknis, yang mencakup topik-topik seperti arsitektur, keamanan, dan ekonomi dan ditulis oleh AWS Solutions Architects atau pakar teknis lainnya.
- AWS Support Pusat — Hub untuk membuat dan mengelola AWS Support kasus Anda. Juga termasuk tautan ke sumber daya bermanfaat lainnya, seperti forum, FAQ teknis, status kesehatan layanan, dan. AWS Trusted Advisor
- AWS Support— Halaman web utama untuk informasi tentang AWS Support, saluran dukungan respons cepat untuk membantu Anda membangun dan menjalankan aplikasi di cloud. one-on-one
- Hubungi Kami – Titik kontak pusat untuk pertanyaan tentang tandaihan AWS , akun, peristiwa, penyalahgunaan, dan masalah lainnya.
- AWS Ketentuan Situs — Informasi terperinci tentang hak cipta dan merek dagang kami; akun, lisensi, dan akses situs Anda; dan topik lainnya.

Memanggil permintaan HTTP kueri IAM API menggunakan

Daftar Isi

Anda dapat mengakses IAM dan AWS STS layanan secara terprogram menggunakan Query. API APIPermintaan kueri adalah HTTPS permintaan yang harus berisi `Action` parameter untuk menunjukkan tindakan yang akan dilakukan. IAMdan AWS STS dukungan GET dan POST permintaan untuk semua tindakan. Artinya, API tidak mengharuskan Anda untuk menggunakan GET untuk beberapa tindakan dan POST untuk orang lain. Namun, GET permintaan tunduk pada ukuran batasanURL; meskipun batas ini bergantung pada browser, batas tipikal adalah 2048 byte. Oleh karena itu, untuk API permintaan Kueri yang memerlukan ukuran lebih besar, Anda harus menggunakan POST permintaan.

Responsnya adalah XML dokumen. Untuk detail tentang respons, lihat halaman tindakan individual di IAMAPIReferensi atau AWS Security Token Service APIReferensi.

> **Tip**
>
> Alih-alih melakukan panggilan langsung ke IAM atau AWS STS API operasi, Anda dapat menggunakan salah satu dari AWS SDKs. AWS SDKsTerdiri dari perpustakaan dan kode sampel untuk berbagai bahasa pemrograman dan platform (Java, Ruby,. NET, iOS, Android, dll.). SDKsMenyediakan cara yang nyaman untuk membuat akses terprogram ke IAM dan AWS. Misalnya, SDKs mengurus tugas-tugas seperti menandatangani permintaan secara kriptografi (lihat di bawah), mengelola kesalahan, dan mencoba ulang permintaan secara otomatis. Untuk informasi tentang AWS SDKs, termasuk cara mengunduh dan menginstalnya, lihat halaman Alat untuk Amazon Web Services.

Untuk detail tentang API tindakan dan kesalahan, lihat IAMAPIReferensi atau AWS Security Token Service APIReferensi.

Titik akhir

IAMdan AWS STS masing-masing memiliki satu titik akhir global:

- (IAM) https://iam.amazonaws.com
- (AWS STS) https://sts.amazonaws.com

> **Important**
>
> AWS STS juga mendukung pengiriman permintaan ke titik akhir regional selain titik akhir global. AWS merekomendasikan penggunaan titik akhir regional alih-alih titik akhir global untuk mengurangi latensi, membangun redundansi, dan meningkatkan validitas token sesi. Sebelum Anda dapat menggunakan AWS STS di Wilayah, Anda harus terlebih dahulu mengaktifkan STS di Wilayah itu untuk Anda Akun AWS. Untuk informasi selengkapnya tentang mengaktifkan Wilayah tambahan AWS STS, lihatKelola AWS STS dalam sebuah Wilayah AWS.

Untuk informasi selengkapnya tentang AWS titik akhir dan Wilayah untuk semua layanan, lihat Titik akhir dan kuota layanan di. Referensi Umum AWS

HTTPSdiperlukan

Karena Query API mengembalikan informasi sensitif seperti kredensi keamanan, Anda harus menggunakan HTTPS dengan semua API permintaan.

IAMAPIPermintaan penandatanganan

Permintaan harus ditandatangani menggunakan access key ID dan secret access key. Kami sangat menyarankan agar Anda tidak menggunakan Pengguna root akun AWS kredensil Anda untuk pekerjaan sehari-hari dengan. IAM Anda dapat menggunakan kredensil untuk IAM pengguna atau Anda dapat menggunakan AWS STS untuk menghasilkan kredensil keamanan sementara.

Untuk menandatangani API permintaan Anda, sebaiknya gunakan AWS Signature Version 4. Untuk informasi tentang menggunakan Signature Versi 4, buka Proses Penandatanganan Signature Versi 4 di Referensi Umum AWS .

Jika Anda perlu menggunakan Signature Versi 2, informasi tentang menggunakan Signature Versi 2 tersedia di Referensi Umum AWS.

Untuk informasi selengkapnya, lihat berikut ini:

- AWS Kredensial Keamanan. Memberikan informasi umum tentang jenis kredensil yang digunakan untuk mengakses. AWS
- Praktik terbaik keamanan di IAM. Menyajikan daftar saran untuk menggunakan IAM layanan untuk membantu mengamankan AWS sumber daya Anda.
- Kredensi keamanan sementara di IAM. Menjelaskan cara membuat dan menggunakan kredensial keamanan sementara.

Riwayat dokumen untuk IAM

Tabel berikut menjelaskan pembaruan dokumentasi utama untukIAM.

Perubahan	Deskripsi	Tanggal
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAMAccess Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang kebijakan IAM pengguna dan peran ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	30 Mei 2024
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAMAccess Analyzer menambahkan dukungan untuk izin untuk mengambil status saat ini dari blokir akses publik untuk EC2 snapshot Amazon ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	23 Januari 2024
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAMAccess Analyzer menambahkan aliran dan tabel DynamoDB ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	Januari 11, 2024
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAMAccess Analyzer menambahkan bucket direktori Amazon S3 ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	1 Desember 2023

IAMAccessAnalyzerReadOnlyAccess — Izin Ditambahkan	IAMAccess Analyzer menambahkan izin IAMAccessAnalyzerReadOnlyAccessuntuk memungkinkan Anda memeriksa apakah pembaruan kebijakan Anda memberikan akses tambahan. Izin ini diperlukan oleh IAM Access Analyzer untuk melakukan pemeriksaan kebijakan pada kebijakan Anda.	26 November 2023
IAMAccess Analyzer menambahkan penganalisis akses yang tidak digunakan	IAMAccess Analyzer menyederhanakan pemeriksaan akses yang tidak digunakan untuk memandu Anda menuju hak istimewa yang paling sedikit. IAMAccess Analyzer terus menganalisis akun Anda untuk mengidentifikasi akses yang tidak digunakan dan membuat dasbor terpusat dengan temuan.	26 November 2023
IAMAccess Analyzer menambahkan pemeriksaan kebijakan khusus	IAMAccess Analyzer sekarang menyediakan pemeriksaan kebijakan khusus untuk memvalidasi bahwa IAM kebijakan mematuhi standar keamanan Anda sebelum penerapan.	26 November 2023

AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAMAccess Analyzer menambahkan IAM tindakan ke izin tingkat layanan AccessAnalyzerServiceRolePolicyuntuk mendukung tindakan berikut: • Entitas daftar untuk kebijakan • Menghasilkan detail layanan yang terakhir diakses • Daftar informasi kunci akses	26 November 2023
Tindakan terakhir mengakses informasi dan dukungan pembuatan kebijakan untuk lebih dari 60 layanan dan tindakan tambahan	IAMsekarang mendukung tindakan informasi yang diakses terakhir dan menghasilkan kebijakan dengan informasi tingkat tindakan untuk lebih dari 60 layanan tambahan, bersama dengan daftar tindakan yang informasi terakhir diakses tindakan yang tersedia.	1 November 2023
Tindakan terakhir mengakses dukungan informasi untuk lebih dari 140 layanan	IAMsekarang menyediakan informasi tindakan yang terakhir diakses untuk lebih dari 140 layanan, bersama dengan daftar tindakan yang informasi terakhir diakses tindakan yang tersedia.	14 September 2023

Support untuk beberapa perangkat otentikasi multi-faktor (MFA) untuk pengguna root dan pengguna IAM	Sekarang Anda dapat menambahkan hingga delapan MFA perangkat per pengguna, termasuk kunci FIDO keamanan, kata sandi satu kali berbasis waktu perangkat lunak (TOTP) dengan aplikasi otentikator virtual, atau token perangkat keras. TOTP	16 November 2022
IAMAkses dukungan Analyzer untuk jenis sumber daya baru	IAMAccess Analyzer menambahkan dukungan untuk jenis sumber daya berikut: • Cuplikan EBS volume Amazon • ECRRepositori Amazon • Sistem EFS file Amazon • Cuplikan Amazon RDS DB • Cuplikan kluster Amazon RDS DB • SNSTopik Amazon	25 Oktober 2022
U2F penghentian dan/perbarui WebAuthn FIDO	Menghapus menyebutkan U2F sebagai MFA opsi dan menambahkan informasi tentang WebAuthn,FIDO2, dan FIDO kunci keamanan.	31 Mei 2022

Pembaruan ketahanan di IAM	Menambahkan informasi tentang mempertahankan akses ke IAM kredensil ketika suatu peristiwa menggangg u komunikasi antara. Wilayah AWS	Mei 16, 2022
Kunci kondisi global baru untuk sumber daya	Sekarang Anda dapat mengontrol akses ke sumber daya berdasarkan akun, Unit Organisasi (OU), atau organisasi AWS Organizat ions yang berisi sumber daya Anda. Anda dapat menggunak an kunci kondisi `aws:Resou rceAccount` `aws:Resou rceOrgID` ,, dan `aws:Resou rceOrgPaths` global dalam IAM kebijakan.	27 April 2022
Contoh kode untuk IAM menggunakan AWS SDKs	Contoh kode yang ditambahk an yang menunjukkan cara menggunakan IAM kit pengembangan AWS perangkat lunak (SDK). Contoh dibagi menjadi kutipan kode yang menunjukkan cara memanggil fungsi layanan individual dan contoh yang menunjukkan cara menyelesa ikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.	7 April 2022

Pembaruan diagram alir logika evaluasi kebijakan	Pembaruan pada diagram alir logika evaluasi kebijakan dan teks terkait dalam Menentukan apakah permintaan diizinkan atau ditolak dalam bagian akun.	17 November 2021
Pembaruan praktik terbaik keamanan	Menambahkan informasi tentang membuat pengguna administratif alih-alih menggunakan kredensi pengguna root, menghapus praktik terbaik menggunakan IAM grup untuk menetapkan izin kepada IAM pengguna, dan mengklarifikasi kapan harus menggunakan kebijakan terkelola alih-alih kebijakan sebaris.	5 Oktober 2021
Pembaruan topik logika evaluasi kebijakan untuk kebijakan berbasis sumber daya	Menambahkan informasi tentang dampak kebijakan berbasis sumber daya dan jenis pokok yang berbeda di akun yang sama.	5 Oktober 2021
Pembaruan untuk kunci kondisi bernilai tunggal dan multivaluasi	Perbedaan antara kunci kondisi bernilai tunggal dan multivalued sekarang dijelaskan secara lebih rinci. Jenis nilai ditambahkan ke setiap kunci konteks kondisi AWS global.	30 September 2021

IAMAccess Analyzer mendukung Titik Akses Multi-Wilayah Amazon S3	IAMAccess Analyzer mengidentifikasi bucket Amazon S3 yang memungkinkan akses publik dan lintas akun, termasuk yang menggunakan Titik Akses Multi-Wilayah Amazon S3.	2 September 2021
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	IAMAccess Analyzer memperbarui kebijakan AWS terkelola yang ada.	2 September 2021
Lebih banyak layanan yang didukung untuk pembuatan kebijakan tingkat tindakan	IAMAccess Analyzer dapat menghasilkan IAM kebijakan dengan informasi aktivitas akses tingkat tindakan untuk layanan tambahan. AWS	Agustus 24, 2021
Menghasilkan IAM kebijakan untuk jejak lintas akun	Sekarang Anda dapat menggunakan IAM Access Analyzer untuk membuat kebijakan berbutir halus berdasarkan aktivitas akses Anda menggunakan AWS CloudTrail jejak di akun lain, misalnya, jejak terpusat. AWS Organizations	18 Agustus 2021

Pemeriksaan kebijakan IAM Access Analyzer tambahan	IAMAccess Analyzer memperpanjang validasi kebijakan dengan menambahkan pemeriksaan kebijakan baru yang memvalidasi kondisi yang disertakan dalam kebijakan. IAM Pemeriksaan ini menganalisis blok syarat dalam pernyataan kebijakan Anda dan melaporkan peringatan, kesalahan, dan saran keamanan beserta rekomendasi yang dapat ditindaklanjuti. IAMAccess Analyzer menambahkan pemeriksaan kebijakan berikut: • Kesalahan - Format utama layanan tidak valid • Kesalahan - Kunci tag tidak ada dalam kondisi • Peringatan Keamanan - Tolak NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan • Peringatan Keamanan - Tolak dengan kunci kondisi tag yang tidak didukung untuk layanan • Peringatan Keamanan - Kunci kondisi berpasangan tidak ada • Saran - Izinkan NotAction dengan kunci kondisi tag	29 Juni 2021

	yang tidak didukung untuk layanan • Saran - Izinkan dengan kunci kondisi tag yang tidak didukung untuk layanan	
Tindakan terakhir diakses dukungan untuk lebih banyak layanan	Anda sekarang dapat melihat tindakan informasi yang terakhir diakses di IAM konsol tentang terakhir kali IAM prinsipal menggunakan tindakan untuk layanan berikut: Tindakan pengelolaan AmazonEC2,IAM, Lambda, dan Amazon S3. Anda juga dapat menggunakan AWS CLI atau AWS API untuk mengambil laporan data. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu sehingga Anda dapat menyempurnakan IAM kebijakan Anda agar lebih mematuhi prinsip hak istimewa yang paling rendah.	19 April 2021
Memantau dan mengendalikan tindakan yang diambil dengan peran yang diasumsikan	Administrator dapat mengonfigurasi IAM peran untuk mengharuskan identitas melewati identitas sumber, yang masuk. AWS CloudTrail Meninjau informasi identitas sumber membantu administrator menentukan siapa atau apa yang dilakukan tindakan dengan sesi peran diasumsikan.	13 April, 2021

Menghasilkan IAM kebijakan berdasarkan aktivitas akses	Sekarang Anda dapat menggunakan IAM Access Analyzer untuk membuat kebijakan berbutir halus berdasarkan aktivitas akses yang Anda temukan di. AWS CloudTrail	7 April 2021
IAMPemeriksaan kebijakan Access Analyzer	IAMAccess Analyzer sekarang menyediakan lebih dari 100 pemeriksaan kebijakan dengan rekomendasi yang dapat ditindaklanjuti selama pembuatan kebijakan.	16 Maret 2021
Opsi validasi kebijakan yang diperluas	Validasi kebijakan yang diperluas tersedia di IAM konsol AWS API, dan AWS CLI menggunakan pemeriksa an kebijakan di IAM Access Analyzer untuk membantu Anda membuat kebijakan yang aman dan fungsionalJSON.	15 Maret 2021
Sumber daya penandaan IAM	Anda sekarang dapat menandai IAM sumber daya tambahan menggunakan pasangan nilai kunci tag.	11 Februari 2021
Kebijakan kata sandi default untuk IAM pengguna	Jika Anda tidak menetapkan kebijakan kata sandi khusus untuk Anda Akun AWS, kata sandi IAM pengguna sekarang harus memenuhi kebijakan AWS kata sandi default.	18 November 2020

Halaman tindakan, sumber daya, dan kunci kondisi untuk AWS layanan telah dipindahkan	Setiap AWS layanan dapat menentukan tindakan, sumber daya, dan kunci konteks kondisi untuk digunakan dalam IAM kebijakan. Anda sekarang dapat menemukan daftar AWS layanan dan tindakan, sumber daya, dan kunci konteks kondisinya di Referensi Otorisasi Layanan.	16 November 2020
IAMpengguna durasi sesi peran yang lebih lama	IAMpengguna sekarang dapat memiliki durasi sesi peran yang lebih lama saat beralih peran di AWS Management Console, mengurangi interupsi karena kedaluwarsa sesi. Pengguna diberikan durasi sesi maksimum yang ditetapkan untuk peran tersebut, atau sisa waktu dalam sesi IAM pengguna, mana yang kurang.	24 Juli 2020
Gunakan Service Quotas untuk meminta peningkatan cepat untuk entitas IAM	Anda dapat meminta peningkatan kuota untuk kuota yang dapat disesuaikan menggunakan IAM konsol Service Quotas. Sekarang, beberapa peningkatan disetujui secara otomatis di Kuota Layanan dan tersedia di akun Anda dalam beberapa menit. Permintaan yang lebih besar dikirimkan ke AWS Support.	25 Juni 2020

Informasi terakhir yang diakses IAM sekarang mencakup tindakan manajemen Amazon S3	Selain layanan informasi yang terakhir diakses, Anda sekarang dapat melihat informasi di IAM konsol tentang terakhir kali IAM prinsipal menggunakan tindakan Amazon S3. Anda juga dapat menggunakan AWS CLI atau AWS API untuk mengambil laporan data. Laporan ini mencakup informasi tentang layanan dan tindakan yang diizinkan yang terakhir kali dilakukan untuk mengakses dan kapan dilakukan oleh penanggung jawab tersebut. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu sehingga Anda dapat menyempurnakan IAM kebijakan Anda agar lebih mematuhi prinsip hak istimewa yang paling rendah.	3 Juni 2020
Penambahan Bab Keamanan	Bab keamanan membantu Anda memahami cara mengonfigurasi IAM dan AWS STS memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan IAM sumber daya Anda.	29 April 2020

sts: RoleSessionName	Sekarang Anda dapat menyusun kebijakan yang memberikan izin berdasarkan nama sesi yang ditentukan oleh penanggung jawab saat mengambil peran.	21 April 2020
AWS pembaruan halaman masuk	Saat masuk di halaman AWS masuk utama, Anda tidak dapat memilih untuk masuk sebagai pengguna Pengguna root akun AWS atau IAM pengguna. Ketika Anda melakukannya, label pada halaman menunjukkan apakah Anda harus memberikan alamat email pengguna root Anda atau informasi IAM pengguna Anda. Dokumentasi ini mencakup tangkapan layar yang diperbarui untuk membantu Anda memahami halaman masuk AWS .	4 Maret 2020

AWS: v iaAWSService dan aws: tombol kondisi CalledVia	Anda sekarang dapat menulis kebijakan untuk membatasi apakah layanan dapat membuat permintaan atas nama IAM kepala sekolah (pengguna atau peran). Saat penanggung jawab mengajukan permintaan ke layanan AWS , layanan tersebut mungkin menggunakan kredensial penanggung jawab untuk melakukan permintaan selanjutnya ke layanan lain. Gunakan `aws:ViaAWSService` kunci kondisi yang harus dicocokkan jika layanan apa pun membuat permintaan menggunakan kredensial penanggung jawab. Gunakan `aws:CalledVia` kunci kondisi yang harus dicocokkan jika layanan tertentu membuat permintaan menggunakan kredensial penanggung jawab.	20 Februari 2020
Simulator kebijakan menambahkan dukungan untuk batas izin	Anda sekarang dapat menguji efek batas izin pada IAM entitas dengan simulator IAM kebijakan.	23 Januari 2020

Evaluasi kebijakan lintas akun	Anda sekarang dapat mempelajari cara AWS mengevaluasi kebijakan untuk akses lintas akun. Hal ini terjadi ketika sumber daya di akun tepercaya termasuk kebijakan berbasis sumber daya yang memungkinkan penanggung jawab di akun lain mengakses sumber daya tersebut. Permintaan harus diizinkan di kedua akun.	2 Januari 2020
Tag sesi	Sekarang Anda dapat menyertakan tanda saat Anda mengambil peran atau menggabungkan pengguna di AWS STS. Saat Anda melakukan operasi `AssumeRole` atau `GetFederationToken`, Anda dapat lulus tanda sesi sebagai atribut. Ketika Anda melakukan `AssumeRoleWithSAML` atau `AssumeRoleWithWebIdentity` operasi, Anda dapat meneruskan atribut dari identitas perusahaan Anda ke AWS.	22 November 2019

Kontrol akses untuk grup Akun AWS di AWS Organizations	Anda sekarang dapat mereferensikan unit organisasi (OUs) dari AWS Organizations dalam IAM kebijakan. Jika Anda menggunakan Organizations untuk mengatur akun AndaOUs, Anda dapat meminta prinsipal milik OU tertentu sebelum memberikan akses ke sumber daya Anda. Prinsipal termasuk Pengguna root akun AWS, IAM pengguna dan peran. IAM Untuk melakukannya, tentukan jalur OU di kunci kondisi `aws:PrincipalOrgPaths` di kebijakan Anda.	20 November 2019
Peran terakhir digunakan	Sekarang Anda dapat melihat tanggal, waktu, dan Wilayah tempat peran terakhir digunakan. Informasi ini juga membantu Anda mengidentifikasi peran yang tidak digunakan di akun Anda. Anda dapat menggunakan AWS Management Console, AWS CLI dan AWS API untuk melihat informasi tentang kapan peran terakhir digunakan.	19 November 2019

Perbarui ke halaman kunci konteks kondisi global	Sekarang Anda dapat mempelajari kapan setiap kunci kondisi global disertakan dalam konteks permintaan. Anda juga dapat menavigasi ke setiap tombol dengan lebih mudah menggunakan daftar isi halaman (TOC). Informasi di laman tersebut membantu Anda menulis kebijakan yang lebih akurat. Misalnya, jika karyawan Anda menggunakan federasi dengan IAM peran, Anda harus menggunakan `aws:userId` kunci dan bukan `aws:userName` kuncinya. `aws:userName` Kuncinya hanya berlaku untuk IAM pengguna dan bukan peran.	6 Oktober 2019

ABACdi AWS	Pelajari cara kerja kontrol akses berbasis atribut (ABAC) dalam AWS menggunakan tag, dan bagaimana perbandingannya dengan model otorisasi tradisional AWS . Gunakan ABAC tutorial untuk mempelajari cara membuat dan menguji kebijakan yang memungkinkan IAM peran dengan tag utama mengakses sumber daya dengan tag yang cocok. Strategi ini memungkinkan individu untuk melihat atau mengedit hanya AWS sumber daya yang diperlukan untuk pekerjaan mereka.	3 Oktober 2019
AWS STS GetAccessKeyInfo operasi	Anda dapat meninjau kunci AWS akses dalam kode Anda untuk menentukan apakah kunci tersebut berasal dari akun yang Anda miliki. Anda dapat meneruskan ID kunci akses menggunakan `aws sts get-access-key-info` AWS CLI perintah atau `GetAccessKeyInfo` AWS APIoperasi.	24 Juli 2019

Layanan Viewing Organizations terakhir mengakses informasi di IAM	Anda sekarang dapat melihat informasi layanan yang terakhir diakses untuk AWS Organizations entitas atau kebijakan di AWS Organizationsbagian IAM konsol. Anda juga dapat menggunakan AWS CLI atau AWS API untuk mengambil laporan data. Data ini mencakup informasi tentang layanan yang diizinkan yang terakhir kali dicoba diakses dan kapan oleh penanggung jawab di akun Organizations. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu, sehingga Anda dapat memperbaiki kebijakan Organizations Anda agar dapat mematuhi prinsip hak istimewa sekecil mungkin.	20 Juni 2019
Menggunakan kebijakan terkelola sebagai kebijakan sesi	Anda sekarang dapat meneruskan hingga 10 kebijakan terkelola ARNs saat Anda mengambil peran. Hal ini memungkinkan Anda untuk membatasi izin kredensial sementara peran.	7 Mei 2019

AWS STS Kompatibilitas wilayah token sesi untuk titik akhir global	Sekarang, Anda dapat memilih apakah menggunakan token titik akhir global versi 1 atau versi 2. Token versi 1 hanya berlaku di AWS Wilayah yang tersedia secara default. Token ini tidak akan bekerja di Wilayah yang diaktifkan secara manual, seperti Asia Pacific (Hong Kong). Token Versi 2 valid di semua Wilayah. Namun, token versi 2 lebih panjang dan mungkin memengaruhi sistem tempat Anda menyimpan token untuk sementara waktu.	26 April 2019
Izinkan mengaktifkan dan menonaktifkan wilayah AWS	Sekarang Anda dapat membuat kebijakan yang memungkinkan administrator untuk mengaktifkan dan menonaktifkan Wilayah Asia Pasifik (Hong Kong) (ap-east-1).	24 April 2019

IAMpengguna halaman kredensyal keamanan saya	IAMpengguna sekarang dapat mengelola semua kredensia lnya sendiri di halaman My Security Credentials. AWS Management Console Halaman ini menampilkan informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengedit kata sandi mereka sendiri, kunci akses, sertifikat X.509, SSH kunci, dan kredensi Git.	24 Januari 2019
Penasihat akses API	Anda sekarang dapat menggunakan AWS CLI dan AWS API untuk melihat layanan informasi yang terakhir diakses.	7 Desember 2018
Menandai IAM pengguna dan peran	Anda sekarang dapat menggunakan IAM tag untuk menambahkan atribut kustom ke identitas (IAMpengguna atau peran) menggunakan pasangan nilai kunci tag. Anda juga dapat menggunakan tag untuk mengendalikan akses identitas ke sumber daya atau untuk mengontrol tag apa yang dapat dipasang ke sebuah identitas.	14 November 2018

Kunci keamanan U2F	Anda sekarang dapat menggunakan kunci keamanan U2F sebagai opsi otentikasi multi-faktor (MFA) saat masuk ke file. AWS Management Console	25 September 2018
Dukungan untuk titik VPC akhir Amazon	Anda sekarang dapat membuat koneksi pribadi antara Anda VPC dan AWS STS di Wilayah AS Barat (Oregon).	31 Juli 2018
Batas izin	Fitur baru membuatnya lebih mudah untuk memberikan karyawan tepercaya kemampuan untuk mengelola IAM izin tanpa juga memberikan akses IAM administratif penuh.	12 Juli 2018
aws: PrincipalOrg ID	Kunci kondisi baru menyediakan cara yang lebih mudah untuk mengontrol akses ke AWS sumber daya dengan menentukan AWS organisasi IAM kepala sekolah.	17 Mei 2018
aws: RequestedRegion	Kunci kondisi baru menyediakan cara yang lebih mudah untuk menggunakan IAM kebijakan untuk mengontrol akses ke AWS Wilayah.	25 April 2018
Peningkatan durasi sesi untuk IAM peran	IAMPeran sekarang dapat memiliki durasi sesi 12 jam.	28 Maret 2018

Alur kerja pembuatan peran yang diperbarui	Alur kerja baru meningkatkan proses pembuatan hubungan kepercayaan dan melampirkan izin ke peran.	8 September 2017
Akun AWS proses masuk	Pengalaman AWS masuk yang diperbarui memungkin kan pengguna root dan IAM pengguna menggunakan tautan Masuk ke Konsol di halaman AWS Management Console beranda.	25 Agustus 2017
Contoh IAM kebijakan	Pembaruan dokumentasi menampilkan lebih dari 30 kebijakan contoh.	2 Agustus 2017
IAMpraktik terbaik	Informasi yang ditambahkan ke bagian Pengguna di IAM konsol memudahkan untuk mengikuti praktik IAM terbaik.	5 Juli 2017
Sumber daya Auto Scaling	Izin tingkat sumber daya dapat mengontrol akses ke dan izin untuk sumber daya Auto Scaling.	16 Mei 2017
RDSDatabase Amazon untuk My SQL dan Amazon Aurora	Administrator database dapat mengaitkan pengguna database dengan IAM pengguna dan peran dan dengan demikian mengelola akses pengguna ke semua AWS sumber daya dari satu lokasi.	24 April 2017

Peran terkait layanan	Peran terkait layanan menyediakan cara yang lebih mudah dan lebih aman untuk mendelegasikan izin ke layanan. AWS	19 April 2017
Ringkasan kebijakan	Ringkasan kebijakan baru memudahkan untuk memahami izin dalam kebijakan. IAM	23 Maret 2017

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.