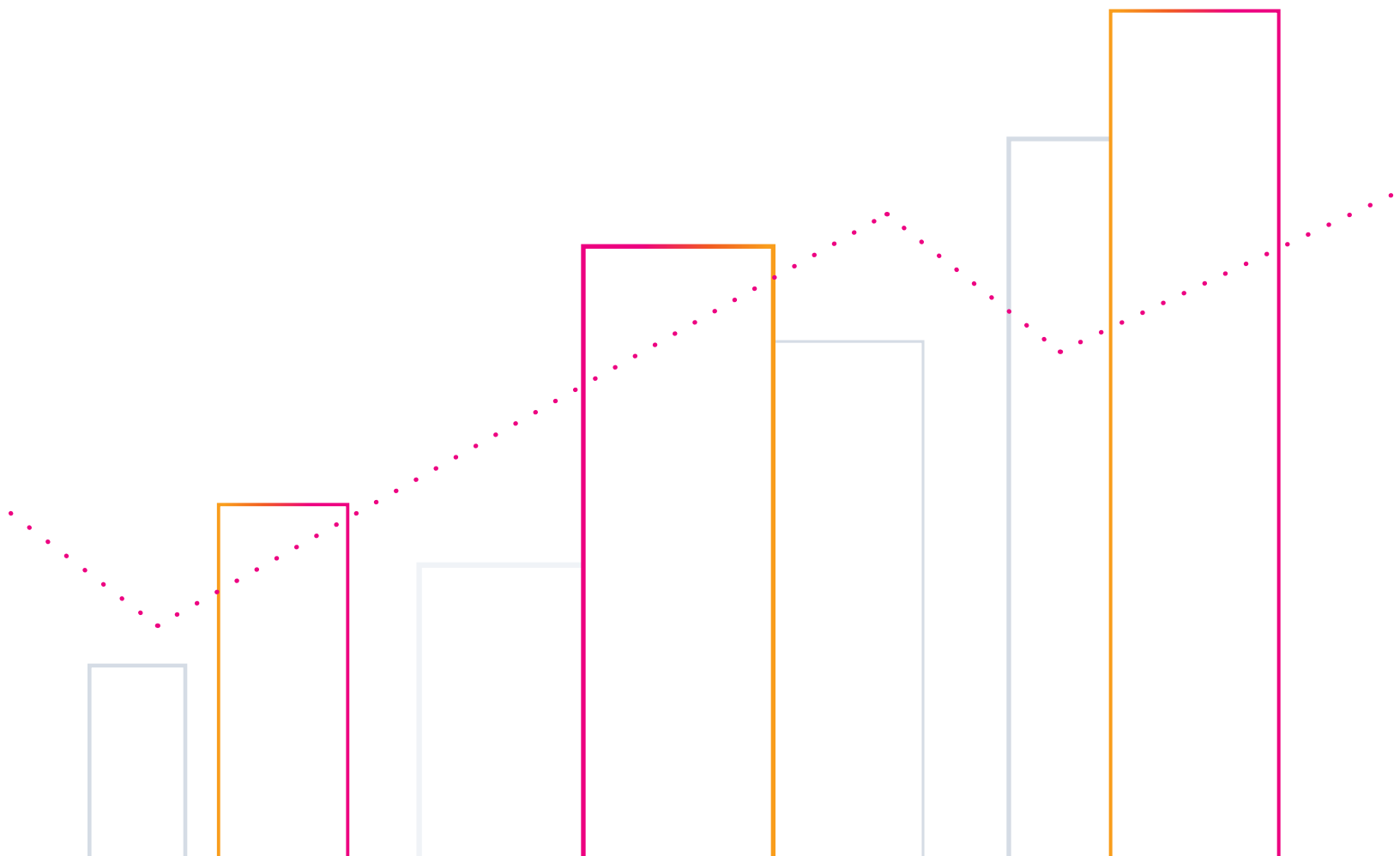


Levelling up your Amazon Connect with Splunk



Overview

In our new world where our customers and employees are connecting from anywhere, it is more important than ever to be able to gain real-time visibility in how our platforms and systems are performing. Not just simple traffic light tracking, I'm talking real, actionable insights - so we get to the 'why' of things faster. The Splunk Data Platform breaks through this challenge by offering users the ability to get actionable insights from your data in real-time.

To showcase this let's double-click on Amazon Connect as a use case. For those who aren't aware, Amazon Connect offers customers the ability to spin up a scalable omnichannel cloud contact center service in just a few clicks.

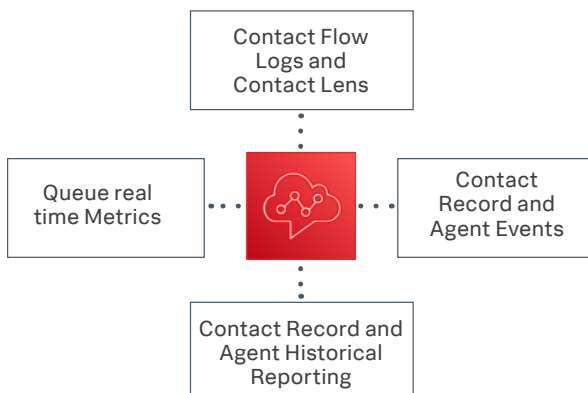
This tech brief will explain how Splunk can be configured to ingest Amazon Connect data to provide valuable insights into how your omnichannel customer service experience platform is working.

Note: This tech brief has been written with one purpose in mind: To allow the reader to configure their Amazon Connect instance to ingest data into Splunk Cloud without the need to open the inevitable one hundred tabs usually required when doing anything like this.

Every deployment and configuration of Amazon Connect and Splunk is different. This paper will try as best as it can to point out best practice vs what is right for your organization.

Amazon Connect Splunk Ingest Points

AWS Connect has several ingest points which will be configured to send data into Splunk. These are outlined below:



Queue Real Time Metrics: Amazon Connect API metrics captures both historic and real time queue metric data from the application's API endpoints.

Contact Flow Logs: Contact flows are used within the AWS Connect product to allow an administrator to define a customer flow experience from start to finish. When an Amazon Connect instance is created an Amazon CloudWatch log group is also created.

Agent Events: Near real time streams report on agent activity within your Amazon Connect instance. These include:

- Agent Login / Logout
- Agent connects with a contact
- Agent status change, such as ability to handle contacts or on a break/ training

Contact Record (CTR) Events: Near real time streams of contact (voice calls, chat, and task) events (for example, call is queued) in your Amazon Connect contact center. These events include:

- Initiated or transferred a voice call, chat or task
- The date or time the customer endpoint connected to AWS Connect
- A voice call, chat, or task is queued to be assigned to an agent.
- A voice call, chat, or task is connected to an agent
- A voice call, chat, or task is disconnected

Contact Record (CTR) and Agent Historical Reporting: Used to capture data about the past, historical reports are stored based on a configured report and schedule. The reports are stored within a S3 bucket for future retrieval.

Contact Lens for Amazon Connect: Helps you follow the sentiment and trends of customer conversations in real time to identify crucial company and product feedback. You can also track the agent compliance of customer conversations in your contact center to ensure standard greetings and sign-offs are used, help train agents, and replicate successful interactions.

Splunk Required Apps and Add-Ons


Throughout this guide we will use several Apps and Add-Ons available free on the [Splunkbase](#).

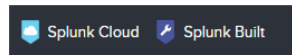
FUN FACT: Do you know the difference between a Splunk Add-On and a Splunk App (or technical add-on, TA)?

The easiest way to think about it is a Splunk App will show you something, e.g. dashboards or a visualization. An add-on will enhance your Splunk functionality.

Required Add-Ons

Splunk Add-On for Amazon Web Services (AWS)

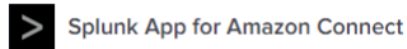
 Splunk Add-on for Amazon Web Services (AWS)



The Splunk add-on for AWS allows Splunk software to collect a whole range of data from AWS by pulling data via authenticated API calls. This add-on is built, maintained and supported by Splunk.

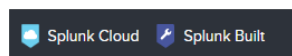
Required Apps

Splunk App for Amazon Connect



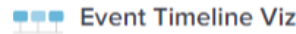
The Splunk App for Amazon Connect is the main app which has pre-configured dashboards and visuals for gaining insights into the performance of your Amazon Connect platform. The App is developed by Splunk and aimed at being an excellent starting point for Amazon Connect customers. This app does not carry standard Splunk support.

Splunk Timeline – Custom Visualization



The Splunk Timeline - Custom Visualization App allows Splunk users to gain better visualization in dashboards and reports. This is required by the Splunk App for Amazon Connect to build our custom visualizations built within the dashboards. Built and Supported by Splunk.

Event Timeline Viz



The Splunk Event Timeline Viz is similar to the Splunk Timeline – Custom Visualization App above allowing Splunk users to get better event timeline visualizations. This is again required by the Splunk App for Amazon Connect in some of the custom dashboard visualizations. This app is not supported by Splunk.

Assumptions and Starting point

As cool as it would be this guide cannot cover everything. Below is a list of some required items.

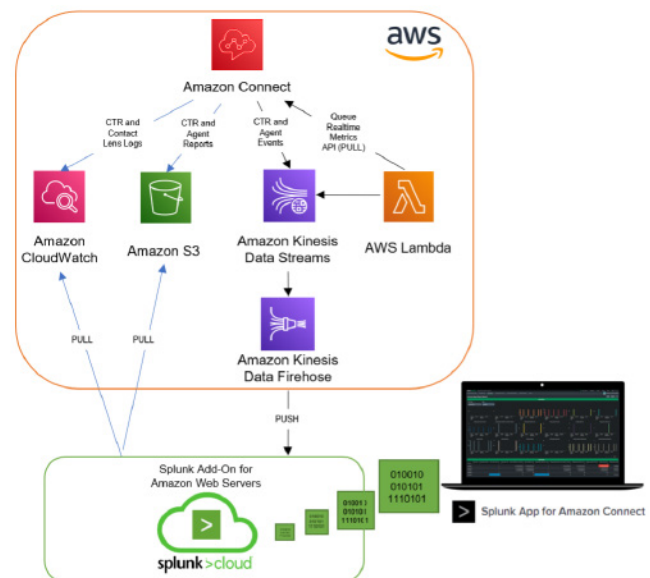
It is also assumed you have the relevant access to AWS Cloud Instance where your Amazon Connect is running as well as your Splunk Cloud Instance.

Required

- Splunk Cloud
- Amazon Connect deployed and configured appropriately within your AWS tenancy

High Level Architecture

Below are the main functional interaction points between AWS and Splunk including data flow direction (pull vs push).



Installing the Splunk Apps / Add-Ons

Below are the steps required to install the Splunk App for Amazon Connect. This technical brief will not go through every app and add-on but the steps are primarily the same.

RESTARTS: Some Add-Ons require a restart of the Splunk Cloud platform. If prompted to do so you can restart either at the very end of installing all the required apps/add-ons or restart when prompted.

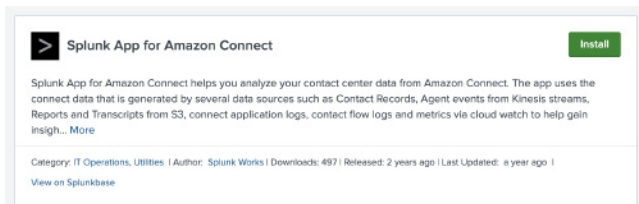
IMPORTANT: Before restarting you should follow your companies change control policies as restarting your Splunk Cloud instance may cause outages or disconnections of the Splunk platform. More information can be found [here](#).

From your Splunk Cloud instance **login** as a user that **has permissions to install applications**.

There are a number of ways you can install applications but for this guide we will **click on the Splunk cloud logo** on the top left corner.

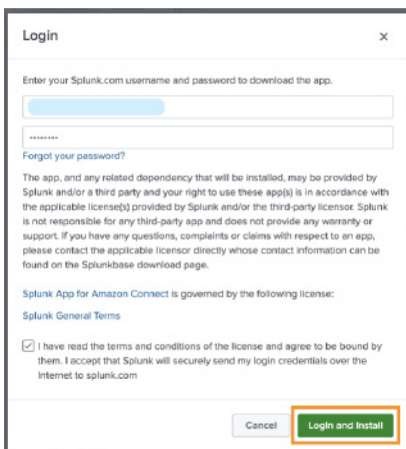


From there you should see **+Find more Apps**. Click it and on this page search for **Connect**, this will return within the results **Splunk App for Amazon Connect** as shown.

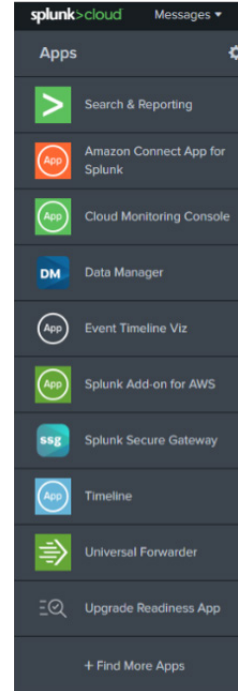


Click **Install**

Enter your **Splunk username** and **password** and **tick the box** agreeing to the terms and conditions.



Once completed it will appear on your list of installed applications. Repeat this process until all of the required apps / add-ons are installed on your Splunk Cloud instance.



- ✓ Splunk Add-On for Amazon Web Services (AWS)
- ✓ Splunk App for Amazon Connect
- ✓ Splunk Timeline – Custom Visualization
- ✓ Event Timeline Viz

Once completed your applications list should look something like the picture shown here.

Amazon Connect Splunk Index

By default, the Splunk App for Amazon Connect creates a new Index for your Amazon Connect ingested data. You will need the name of this index for steps within this document. Alternatively, you can create a brand new index or use an existing one following the additional steps below.

OPTION 1: Get existing index name

From your Splunk console select, **Settings, Indexes**.

Then from the list of indexes look down the **App** column for **amazon_connect_app_for_splunk**. There should be **two**. <indexname> and <indexname>_sum.

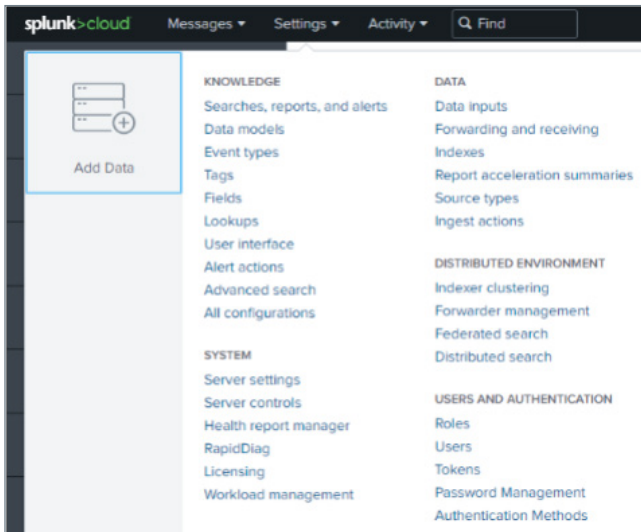
Note down just the index one. Not the one with _sum on the end.

Example below:

Name	Actions	Type	App
aws	Edit Delete Disable	Events	amazon_connect_app_for_splunk
aws_sum	Edit Delete Disable	Events	amazon_connect_app_for_splunk

OPTION 2: Create a new Index

Within your Splunk Cloud console click the **Settings** followed by **Indexes** (under DATA section).



From the Index screen, click **New Index**.

Choose an appropriate **name** for your index.

Select **Events** as your Index type.

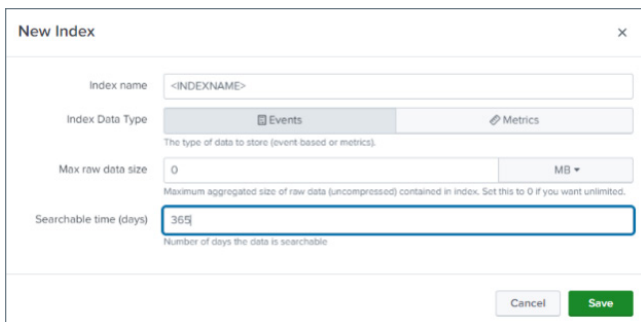
Max raw data size can be set to **0GB** (unlimited) and searchable time (days) to your preferred timeframe.

We are going to use 365 days.

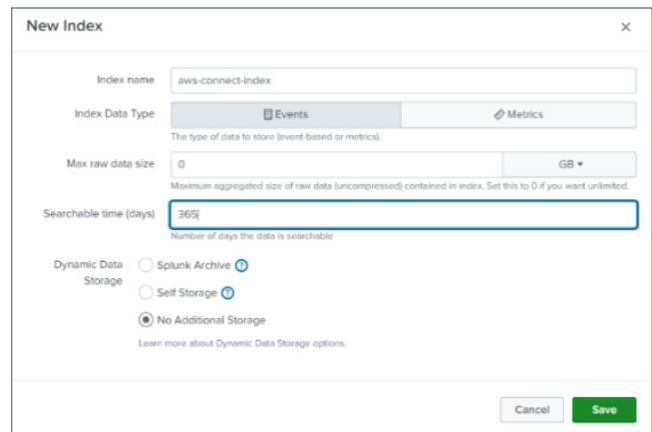
Note: Max raw data size and Searchable time (days) fields as shown in example may vary depending on your organizations policies and procedures in regards to creating Splunk indexes. Adjust as required.

For **Splunk Cloud Victoria** customers you will see an additional option for Dynamic Data Storage (for this example we will select **No Additional Storage**).

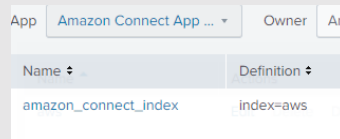
Splunk Classic View



Splunk Victoria



Important Note: If you are using an index other than the default one installed with the App then you will need to update the default search index used by the App by clicking on **Settings, Advanced search, Search Macros** and then click on the **amazon_connect_index** and **replace** the default index=<xxx> with the one you are using. Example below:

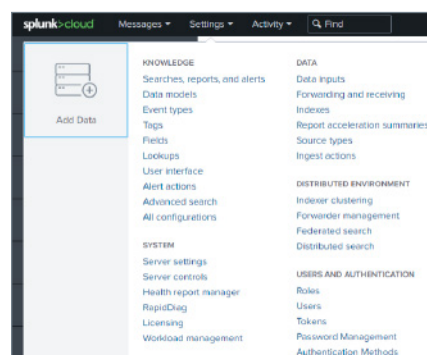


Create a Splunk HEC token

A Splunk HEC (HTTP Event Collector) service is used to collect data from many different sources. In this instance it will be the primary ingestion service for our AWS firehose streaming services.

A Splunk Cloud instance can have many HEC endpoints. We will create a new one specifically for your Amazon Connect data.

Create a HEC endpoint by clicking on **Settings** followed by **Data Inputs** (under the DATA section)



Click **+Add new** on the HTTP Event Collector.

Type	Inputs	Actions
HTTP Event Collector Receive data over HTTP or HTTPS.	1	+ Add new

Enter a **Name** and optional description for your HEC endpoint. For Amazon Connect we recommend leaving **Source name override empty** and **HEC acknowledgment** options **ticked**.

Click **Next** once done.

NOTE: HEC acknowledgment allows the AWS Kinesis firehose to know that the data sent into Splunk has been successfully sent, without this the retry feature of firehose will be invoked. There is also a difference between acknowledgement of data successfully sent vs indexed. For more information on this please see the documentation linked [here](#).

On the **Input Settings** page click the **Select** option under Source type

Select **aws:connect:firehose** as the source type to use for this HEC token

Select the index you created earlier as your **allowed** and **default index's**.

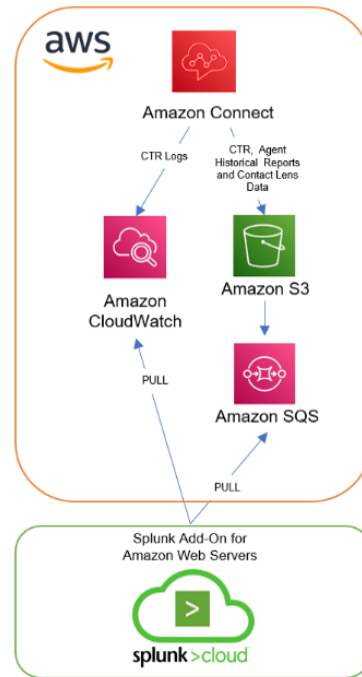
Example below:

Click **Review** to review your Setup. Click **Submit** to create your token when ready.

Copy your HEC token down somewhere for future use. You can find it again later by going back to your Settings, Data Inputs, HEC section.

Amazon Connect IAM Policies

The following AWS IAM policies need to be created in order for the Splunk Add-On for AWS to be authenticated to pull data out of AWS. From our earlier diagram this is relating to the following ingest paths.



Create required policies by searching and clicking on the **IAM** section within your **AWS console**.

Select **Policies** and click **Create Policy**.

Select the **JSON** tab and paste the following policy JSON code into the editor. For more information [see](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":
      "MinPermissionsRequiredToAllowUseOfSQSBased
      S3Actions",
      "Effect": "Allow",
      "Action": [
        "sqs:GetQueueUrl",
        "sqs:ReceiveMessage",
        "sqs:SendMessage",
        "sqs>DeleteMessage",
        "sqs:GetQueueAttributes",
        "sqs:ListQueues",
        "s3:GetObject",
        "kms:Decrypt"
      ],
      "Resource": "*"
    }
  ]
}
```

Click Next: Tags and add any tags required by your organization.

Click Next: Review and enter in a **Name, Description** (optional) and finally **Create policy**.

This policy is specifically for viewing and modifying SQS related activities and the ability to get an object from S3.

Create another policy for reading in Amazon CloudWatch data.

Repeat steps above but this time insert the following JSON code. For more information [see](#).

```
{
  "Statement": [
    {
      "Action": [
        "cloudwatch:List*",
        "cloudwatch:Get*",
        "autoscaling:Describe*",
        "ec2:Describe*",
        "s3:List*",
        "sqs:List*",
        "sns:List*",
        "lambda:List*",
        "elasticloadbalancing:Describe*"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ],
  "Version": "2012-10-17"
}
```

NOTE: Policies listed here are a guide and should be checked by your internal AWS or Security team in order to make sure they comply with your AWS security policies.

Creating an AWS IAM User

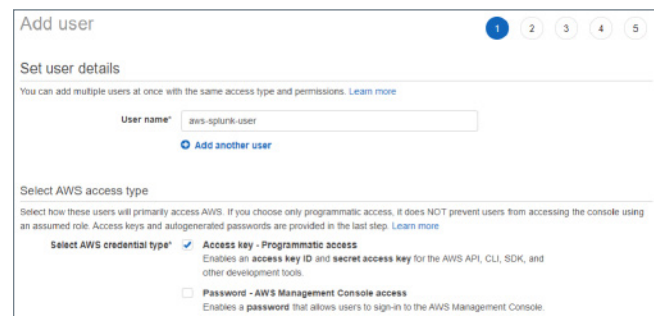
An AWS IAM programmatic user is required for Splunk to authenticate into AWS to pull back data.

This is used by the Splunk Add-on for AWS as shown in the diagram earlier.

Login to your AWS console and enter the **IAM section** of your AWS tenancy.

In IAM section click **Users** followed by **Add User**.

Enter in a **name** for your AWS Splunk user and select the **Access key – Programmatic access** option.

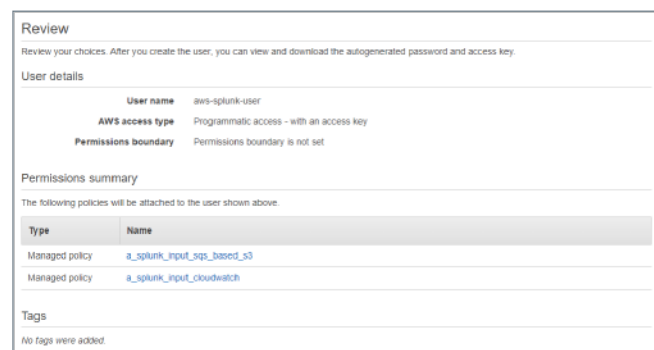


Click Next: Permissions and select the box saying **Attached existing policies directly**.

Select the **policies** you created earlier followed by **Next: Tags**.

Add any tags required by your organization and click **Next: Review** when ready.

Review your user details. Example shown below. Click **Create user** when ready.

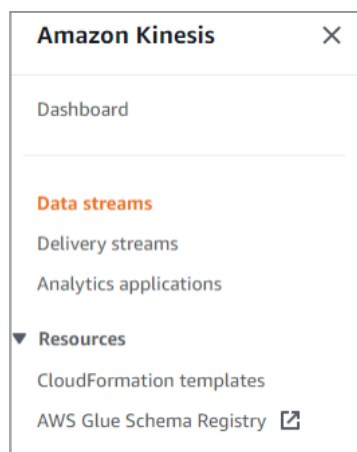


Once user is created either **copy** your **Access key ID** and **Secret access key** or click the **Download .csv** button to download a copy of the new use credentials.

AWS Streams and Firehose Connections

Amazon Connect supports Amazon Kinesis Data Streams as a method of data streaming. In order to stream data into Splunk we need to first create an Amazon Kinesis Data Stream and the relevant Amazon Kinesis Data Firehose connection to Splunk.

From your AWS Console search for **Kinesis Data Streams** and select it from the search list.



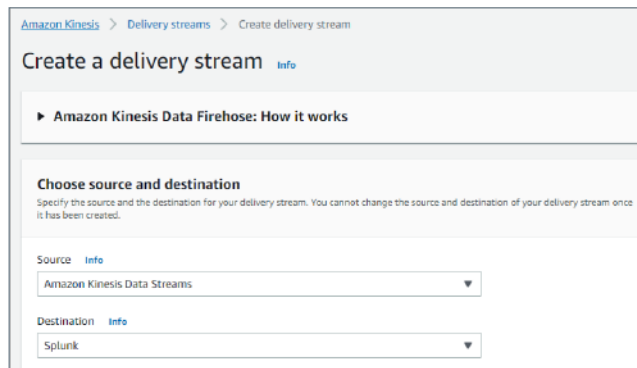
PRO TIP: IAM is a global service whereas a lot of AWS services are regional services. It is important when using services like firehose to check you have selected the correct region first on the top right-hand side of your AWS console. Then deploy your service.

Select **Data streams** from the left-hand side and then click **Create data stream**.

Enter a **Data Stream Name** and then click **Create data stream** leaving capacity as **On-Demand**.

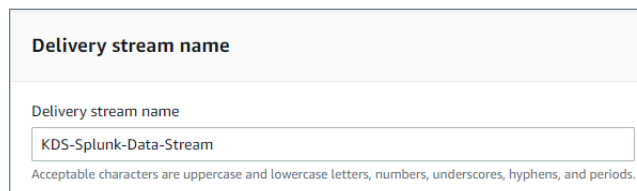
Next Click **Delivery Streams** from the left-hand side followed by **Create data stream button**.

From the Create a delivery stream wizard choose **Amazon Kinesis Data Streams** as the **Source** followed by **Splunk** as the **Destination**. This will then expand out the rest of the options as per picture below.



Browse and **Select** your **Kinesis data stream** you created earlier.

Enter in a **Delivery stream name**.



Leave the **transform records** section as **Disabled**.

NOTE: The Splunk cluster endpoint URL can vary depending on your environment. For instance, a clustered cloud environment (typical customer setup) will use a url of https://http-inputs-<HOSTNAME> on port 443 where as an on-premise or cloud trial account will use a url of https://<HOSTNAME> on port 8088.

A good way to test this would be to run a very basic curl command and check for a successful response. Amazon Kinesis Firehose requires HTTP Event Collector (HEC) endpoint to be terminated with a valid CA-signed certificate matching the DNS hostname used to connect to your HEC endpoint. You must use a trusted CA-signed certificate. Self-signed certificates are not supported. Further information on this can be found [here](#).

Enter in your **Splunk cluster endpoint URL** into the Splunk cluster endpoint section.

Leave the Splunk endpoint type as **Raw endpoint**.

Enter in the **HEC authentication token** you copied earlier into the Authentication token box.

Leave the **timeout settings** as the **default**.

For **Backup settings** select Failed events only and you can then select an existing bucket or choose to create a new S3 bucket for any failed messages.

NOTE: The backup bucket will be used to store any streamed messages that fail to be sent to Splunk HEC. This guide does not cover the setup of the failed messages to be retried directly although the process described within this document to stream SQS to S3 into Splunk using the Add-On for AWS can also be used to pull failed messages into Splunk. Note this would need to be configured or set up with a different Splunk source type depending on your design.

Click **Create Delivery Stream** when ready.

NOTE: Depending on your requirements you could configure a separate kinesis delivery and data streams for each ingest type. In this example we have used the same stream for all three ingest types.

Amazon Connect real time metrics API

The Amazon Connect has a real time metrics API which can be used to pull data from AWS Connect and send to an Amazon Kinesis Data Stream and then through to Splunk. It's a combination of both pull and push.

To do this we need to first configure an AWS Lambda function to run and pull the data from the API and send it to a chosen Amazon Kinesis Data Stream (one we created earlier). More details [here](#).

The AWS Lambda function is built and maintained by Splunk and available via the AWS Lambda console.

To set this up, search and click on **lambda** from the **AWS console**.

From the AWS lambda **select Functions** from the left-hand menu followed by **Create function**.

On the create function page select **Browse serverless app repository** and enter in **Splunk** in the **search box**.

Also tick the box for **Show apps that create custom IAM role**. See example below.

Find an application called **splunk-amazon-connect-api-metrics-processor**. It may be on the second page.

Select this and you will be taken to the next screen to fill in the application details.

Enter an **Application Name**. This can be anything you will use to uniquely identify your application.

Fill in the **Connect Instance ID**. This can be found by clicking on your Amazon Connect instance and grabbing the last remaining characters from your Amazon Connect ARN.

Connect Instance ID Example: `arn:aws:connect:<REGION>:<AWSACCOUNTID>:instance/<CONNECT_INSTANCE_ID>`.

Enter the name of the **Amazon Kinesis Data Stream** you created earlier (The data stream, not the firehose stream). See example below.

NOTE: Before saving, check that you do not have spaces at the start or end of your kinesis data stream when you enter it into Amazon Lambda as described above, this is tricky to troubleshoot later.

Tick the box for **acknowledging** the IAM roles.

Click **Deploy** once ready and once deployed it will be in your AWS Lambda application list.

Configure SQS Queues for Splunk

SQS queues are used in our setup for Splunk to pull data out of Amazon S3 buckets.

Four SQS queues will need to be created in total as we use SQS queues for historical reports and contact lens data each queue also requiring a dead letter queue which is why there are four required. The steps will cover how to create one which you can repeat to create the remaining.

Suggested names are:

- SQS1: **connect-historical-reports**
- SQS2: **connect-historical-reports_DEADLETTER**
- SQS3: **connect-contactlens**
- SQS4: **connect-contactlens_DEADLETTER**

Search and click on the **Simple Queue Service** in the **AWS Console**.

Click on **Create Queue**.

Name the queue and set the **visibility timeout** settings to **5 minutes**.

Under **Access Policy** select the **Advanced** radio button and **copy the existing block of JSON code** into a separate text editor **to be used later**.

Once you have saved off the old JSON code, **replace that JSON code** in the AWS console with this code below modifying the sections in **<xxx>**.

```
{
  "Version": "2012-10-17",
  "Id": "SQS-policy",
  "Statement": [
    {
      "Sid": "s3-principal",
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": "SQS:SendMessage",
      "Resource": "<the ARN of the SQS queue>",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your AWS account number>"
        }
      },
      "ArnLike": {
        "aws:SourceArn": "<the ARN of the Amazon Connect S3 bucket>"
      }
    }
  ]
}
```

NOTE: SQS queue ARN and Account ID will be in the code copied earlier. The AWS Connect S3 bucket ARN can be found by going to the AWS S3 bucket service and selecting your S3 bucket corresponding to your AWS Connect instance. From here the ARN will be under properties.

Leave the rest of the defaults and click **Create Queue** when ready.

Repeat the exact steps you just did for create an SQS queue for the 3 remaining queues adjusting the name to something similar as shown earlier

Once completed you should have **FOUR** SQS queues like the example shown below.

connect-contactlens	Standard
connect-contactlens_DEADLETTER	Standard
connect-historical-reports	Standard
connect-historical-reports_DEADLETTER	Standard

Now we need to **click on the** non-deadletter queues (two of them) and **select the Dead-letter queue tab on each of them.**

Click **edit** and **scroll down to the Dead-letter queue** section **expanding it.**

Select the Enable option and **choose your corresponding dead-letter queue** you created earlier.

Leave retries as is.

Click **Save** once done.

Once you have done both. Make sure you have double checked that the corresponding DEADLETTER queue matches the correct SQS queue you created earlier.

Configure S3 Bucket to trigger SQS

From the AWS console, **search and click on S3.**

Select your **Amazon Connect S3 bucket** followed by the **Properties tab.**

Scroll down to the section **Event Notifications** and click the **Create event notification.**

In the **General Configuration** choose a name for the event.

Under **Prefix**, set this to the following:
connect/<CALL CENTER NAME>/Reports/

NOTE: Obtain your Call Center Name by searching for your Amazon Connect instance in your AWS Console and copying the name of the call center you are configuring in this guide.

Additional Note: Unless you have already configured and are using Contact Lens / Scheduled reports then actual folders will not exist in your S3 bucket for your call center. This guide will show you how to configure these options below but until these features are active and used (ie reports generated or contact lens calls made) then the folders and corresponding data will not exist.

Set the **Suffix** to be **.csv** as this is the format the reports are written in.

General configuration

Event name

Event name can contain up to 255 characters.

Prefix - optional
 Limit the notifications to objects with key starting with specified characters.

Suffix - optional
 Limit the notifications to objects with key ending with specified characters.

Under Event Types select the **All object creation events** tick box on the left hand side as shown.

Event types
Specify at least one event for which you want to receive notifications. For each group, you can choose an event type for all events, or you can choose one or more individual events.

Object creation

All object create events
s3:ObjectCreated:*

Put
s3:ObjectCreated:Put

Post
s3:ObjectCreated:Post

Copy
s3:ObjectCreated:Copy

Multipart upload completed
s3:ObjectCreated:CompleteMultipartUpload

Leave all other tick boxes empty.

Under Destination select **SQS queue** and then select **Choose from your SQS queues.**

Select the queue you created earlier from the drop-down list. Example below.

Destination

ⓘ Before Amazon S3 can publish messages to a destination, you must grant the Amazon S3 principal the necessary permissions to call the relevant API to publish messages to an SNS topic, an SQS queue, or a Lambda function. [Learn more](#)

Destination
 Choose a destination to publish the event. [Learn more](#)

Lambda function
Run a Lambda function script based on S3 events.

SNS topic
Send notifications to email, SMS, or an HTTP endpoint.

SQS queue
Send notifications to an SQS queue to be read by a server.

Specify SQS queue

Choose from your SQS queues

Enter SQS queue ARN

SQS queue

Click **Save changes** once done.

NOTE: If you see the example error below, then it is most likely in SQS step earlier where you modified the JSON access policy (specifically the Amazon Connect S3 bucket ARN) is incorrect. Check the ARN for your Amazon Connect S3 bucket matches what is in your SQS access policy and try again.



Repeat the exact steps above for **Creating an event notification** but this time create one for the **AWS contact lens information**.

The only difference is the prefix should be **Analysis/Voice/** and the Suffix should be **.json** all other steps including SQS queue should be the same.

Once completed you should see **two event notifications** in your list.

Name	Event types	Filters	Destination type	Destination
splunk_sqs_notification_contacts	All object create events	Analysis/Voice/.json	SQS-queue	connect-contacts-logs
splunk_sqs_notification_reports	All object create events	connect/tes-callcenter/reports/.civ	SQS-queue	connect-tes-reports

Configure Amazon Connect to Stream Data

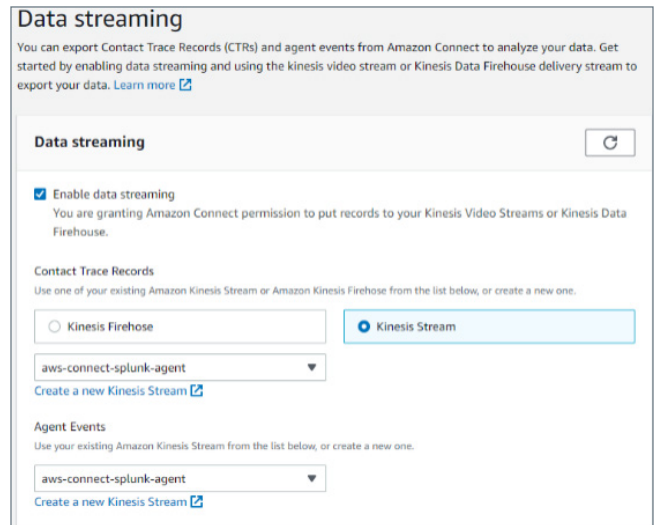
Configure Amazon connect to send agent and CTR data to the AWS data streams.

Search and **select the Amazon connect** from the AWS console. Select your Amazon connect call center.

On the left hand side panel select **Data Streaming**.

Tick the **Enable data streaming** option and **select the Kinesis stream radio button**.

Under both **Contact Trace Records** and **Agent events** **select the kinesis stream you created earlier** from the drop down list. Example below.



Click **Save** when done.

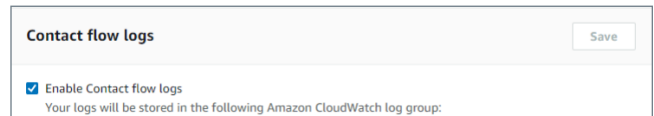
Enable Contact Flow Logs

Search and **select the Amazon Connect** from the AWS console. Select your **Amazon Connect call center**.

On the left-hand panel, **select Contact flows**.

Under Contact flow logs, **tick the Enable Contact flow logs** if not already ticked.

Click **Save** to update the details.



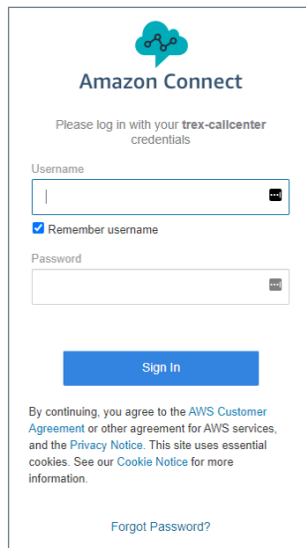
Configure Amazon Connect Contact Lens

Amazon Connect contact lens needs to be enabled within an Amazon Connect contact flow.

Search and **select for Amazon Connect** from your AWS console.

Click on your call center from your list.

From the main overview screen **select the Access url** to open a new tab and **login to your Amazon Connect Call Center**.



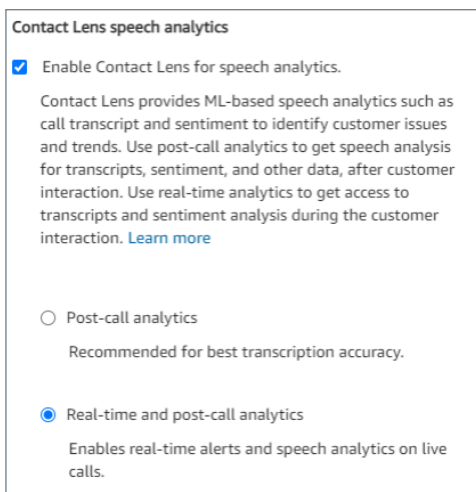
From your Amazon Connect Dashboard on the **left hand side panel** select **Routing** then click the **contact flows menu option**.

Select the contact flow you wish to enable contact lens on.

As part of your contact flow you should have a **Set recording and analytics behavior contact flow box**.

Click this and a **side panel** should appear.

From here, **tick the Enable Contact Lens for speech analytics** followed by the **Real-time and post-call analytics**.



Click **Save** when done.

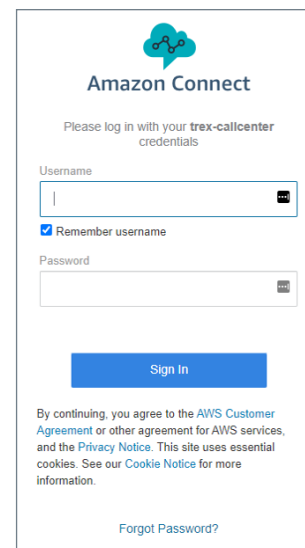
NOTE: If you do not have the Set recording and analytics behavior contact flow box then you will need to add one to get call recordings and contact lens working. Your Amazon Connect team or partner will help you to implement this option appropriately for your call center.

Scheduled Amazon Connect Reports

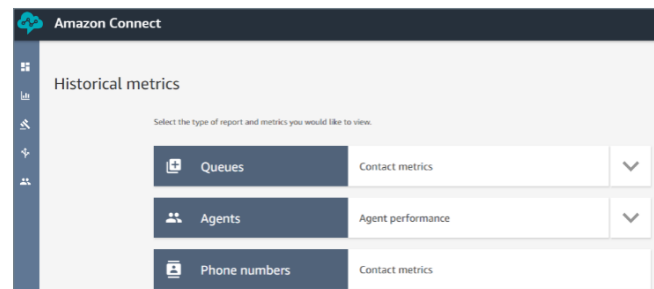
Amazon Connect can run and schedule regular reports in .csv format. These will then be ingested into Splunk via the Splunk Add-On for AWS.

To create and schedule a report login to your Amazon Connect instance by **searching and selecting for Amazon Connect** from your AWS console.

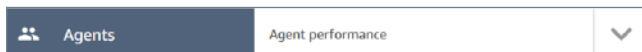
From the Amazon Connect overview screen **select the Amazon Connect instance url** and **login to the Amazon Connect Instance**.



From here select the **Analytics option** from the **left-hand menu** followed by the **Historical metrics** option.



First create an Agent performance report by selecting the **Agent performance box**.



From the **Historical metrics: Agents** section click the **small gray cog** on the right hand side.

Under the **Interval and Time range** click the Time range drop down and select **Today (since 12am)**.

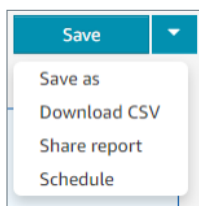
While in the same section **select the drop down for Time Zone** and **select the time zone format** for the report.

NOTE: The time zone selected in this section is used to align the time zone to when the day starts, eg 12am (midnight) in a time zone of your choice. Usually, the time zone you are operating your call center in.

Select the **Metrics Tab** and then proceed to **tick EVERY tick box** making sure you **scroll down** as well. This will collect all metrics.

Click **Apply** to build the report. Once the report has been run we now need to create a schedule for the same report to continue to be run.

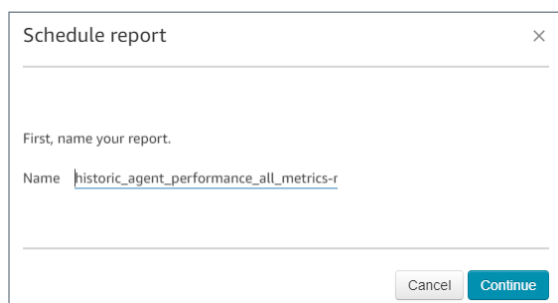
Select the **drop arrow next to the Save button** and click the **Schedule** option.



Name the report example as below **historic_agent_performance_all_metrics-<XX>**.

The <XX> can be replaced with something specific for your organization but the **prefix section must remain the same**.

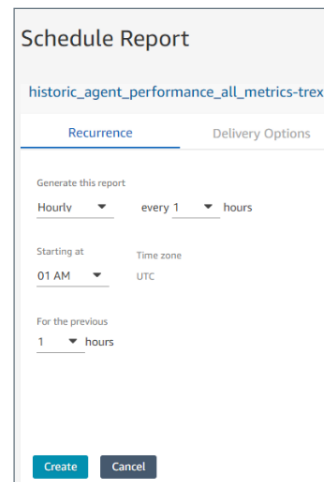
This ensures that the Splunk App for Amazon Connect reporting dashboard works correctly out of the box. Example below.



Click **Continue** once named.

Click **Continue** again to accept the reporting note.

Select the report to be **generated every hour** for the **previous 1 hour**. See example.



Click **Create** once done.

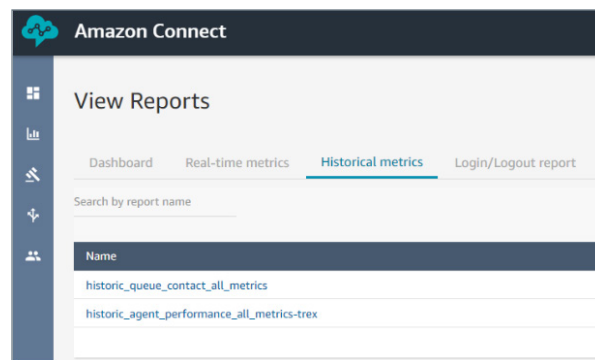
Repeat the steps above for creating and scheduling report but this time **select Queues** instead of Agents instead.



Make sure you name the report **historic_queue_contact_all_metrics-<XX>**.

Once completed, select **Analytics, Saved Reports** option from the left-hand side.

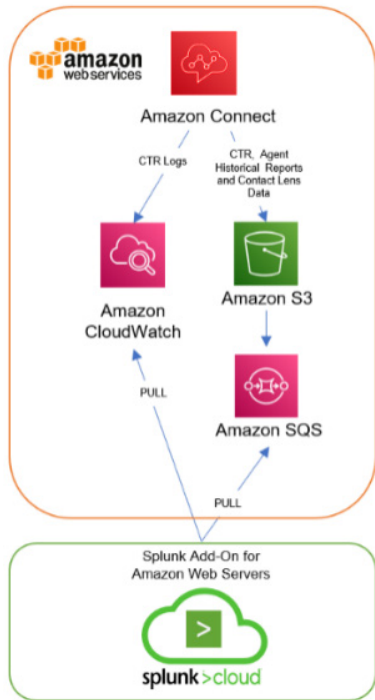
Then select **Historical Reports**. You should see your two reports created. Example below.



NOTE: It may take up to one hour before you start seeing data under this source type due to the Amazon Connect report running each hour. This will then trigger the SQS and Splunk will pull it in.

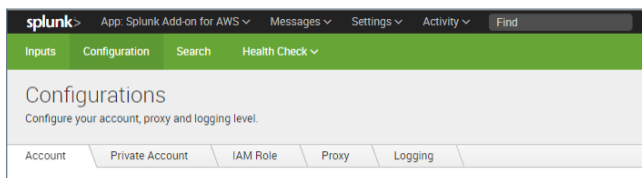
Configure Splunk Add-on for AWS Inputs

Next we need setup the PULL sections from the Splunk Add-on for AWS to ingest our reports as depicted in the diagram below.



Login to Splunk Cloud and from the Apps drop down list select Splunk Add-on for AWS.

Select the Configuration tab from the top section.



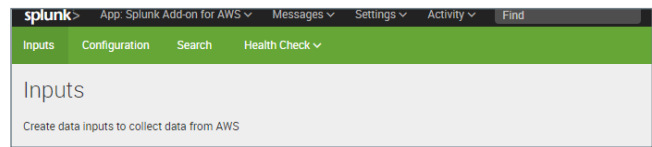
From the right hand side, click the Add button.

Enter a Name (describing the user account) and then copy in your Key ID and Secret key from you excel sheet you downloaded when creating the AWS IAM user earlier in this guide.

Select your Region and click add.

Your username should now appear in the list under configurations.

Next select the Inputs Tab on the top left corner.



From the Create New Input drop down on the right-hand side select Custom Data Type, SQS based S3 from the list.

Name the AWS input something meaningful, eg Historic_Agent_Queue_Reports

Select from the drop down list the AWS Account you configured earlier.

Select your AWS region where your AWS Connect instance is running. This will then populate the list of SQS Queues.

Select the SQS queue you created earlier for the historical reports.

If exists Untick Signature Validate All Events (new feature in Version 6.0 of AWS_TA)

Modify the Source Type name to be aws:connect:s3:reports.

Set Index to be the index you created earlier.

Leave remaining defaults and click Submit once completed.

The input will now be configured.

Repeat the steps above creating another Input of the same type custom SQS based S3 but this time for our contact lens SQS created earlier and using the Source Type aws:contactlens:s3.

Lastly, configure an input for the Amazon Connect CloudWatch logs by selecting the Create New Input button, Customer Data Type, CloudWatch Logs.

Similar to before enter in a name, your AWS account, region where your Amazon Connect instance is running.

For Log Group enter in the following details: /aws/connect/<CALL CENTER NAME>

NOTE: The path for your call center log group can be found by selecting CloudWatch from your AWS console. Then select Log groups from the left hand side. Here you will see your call center log group path.

For Source Type enter in **aws:connect:cloudwatchlogs**.

Set **Index** to be the **index you created earlier**.

Click **Save** to save the input.

Configure CloudWatch Metrics (custom).

In order to extract the Amazon Connect CloudWatch metrics we need to configure a fourth Input in the Splunk Add-on for AWS app.

From the Splunk Add-on for AWS under **Inputs**, select **Create new Input** then **CloudWatch**.

Enter a **name**, eg Connect Instance Metrics and same as before select your **AWS Account**.

Select your **AWS Regions** where your Amazon Connect instance is running.

Choose a **sourcetype** of **aws:connect:cloudwatch:metrics** and **select** the **index your created earlier for Amazon connect**.

NOTE: Under advanced settings you can adjust the polling period. This is the same for all the Splunk Inputs. This is how often Splunk will wait till it attempts to make another pull of the data to get the latest. This figure will vary depending on how often the data on the AWS side is also updated. Adjust as required.

Lastly, we need to configure customer metrics configuration. **Click** the **(Edit in Advanced mode)** section next to Metrics Configuration as shown below.

First thing is to **remove all** the **default Namespace** sections on the left by **clicking** the 'X' on all of them.

Click **+ Add Namespace** and enter a name of **AWS/Connect** (which is the namespace used for Amazon Connect).

On the right-hand side **click + Add Another** which will bring up boxes for Dimension Value, Metrics and Metrics Statistics, **repeat this twice** more to get **three sets of empty boxes up**.

In the first box under **Dimension Value?** enter the text below.

Box1:

```
[{"Participant":[".*"],"Type of Connection":[".*"],"Instance ID":[".*"],"Stream Type":[".*"]}]
```

Once you click away from the box it should **auto populate the Dimension section** to the left of it.

Repeat the same for the two bits of code below in the **second and third box**.

Box 2:

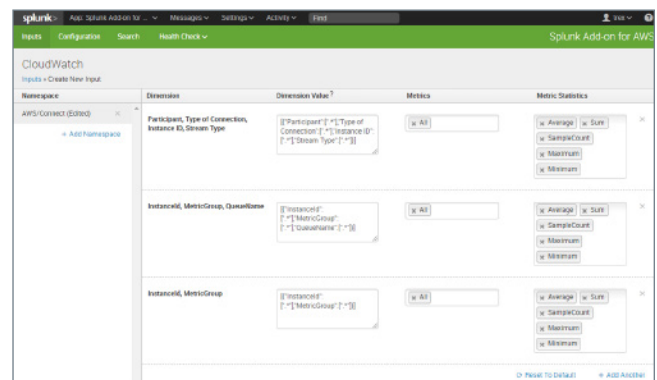
```
[{"InstanceId":[".*"],"MetricGroup":[".*"],"QueueName":[".*"]}]
```

Box 3:

```
[{"InstanceId":[".*"],"MetricGroup":[".*"]}]
```

Next **click inside** the **Metrics box** and select **All** for each of the **three boxes**.

In the **Metrics Statistics** click multiple times **selecting all** the options for **Average, SampleCount, Sum, Maximum and Minimum** for **all three boxes**. Final example shown below.

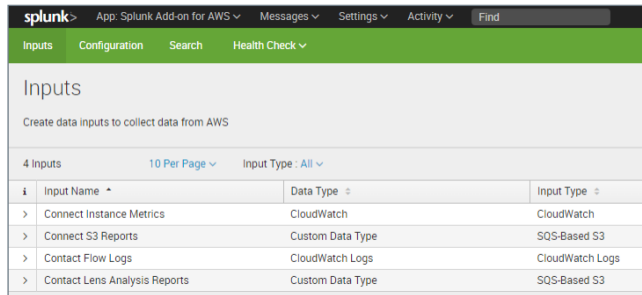


Click **Ok** to save and you should now see it represented on the input page.

Metrics Configuration (Edit in advanced mode)		
Name Service (1)	Dimensions	Metrics
AWS/Connect	Modified	Modified

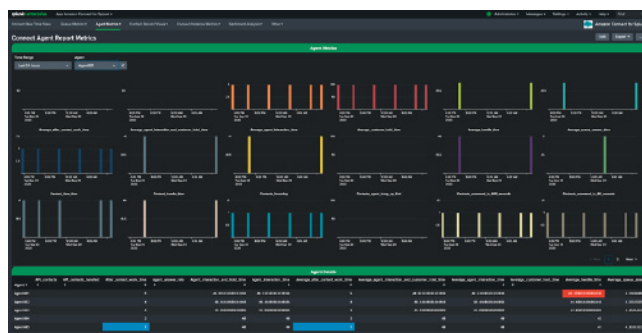
Once completed click **save** to save this input.

Once completed you should now have **four Input types** listed as shown below.



Input Name	Data Type	Input Type
Connect Instance Metrics	CloudWatch	CloudWatch
Connect S3 Reports	Custom Data Type	SQS-Based S3
Contact Flow Logs	CloudWatch Logs	CloudWatch Logs
Contact Lens Analysis Reports	Custom Data Type	SQS-Based S3

CONGRATULATIONS! If you have done all of this correctly then you should now start seeing Amazon Connect data flowing into your **Splunk App for Amazon Connect** dashboards.

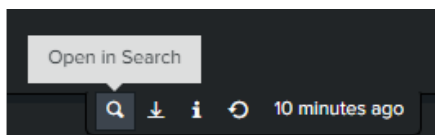


Troubleshooting

Below is some basic go look here statements around not seeing data being ingested into Splunk from your Amazon Connect Instance or particular dashboards which are not correctly loading.

Where to begin?

If you are not seeing a particular set of data you are expecting in a dashboard a good place to start is to check the search results manually. To do this **put your mouse over the dashboard** and **click on the Open in search** option as shown below:



This will show you what the search is doing. You can then start to modify, eg adjust time range, cutting the complexity back to just source type or modify some parameters.

If nothing is coming up in the results of that particular source type then it is most likely something wrong with the configuration.

More Specific Issues with...

Queue or Agent Metrics – Real Time

Requires: [Amazon Data Streams/Firehose](#), [HEC Token](#) and [Lambda setup](#).

Queue or Agent Metrics – Historical

Requires: [SQS](#), [Amazon Connect Reports](#), [Splunk Add-On for AWS](#), [SQS for S3 Input in Add-on](#) and [IAM Account](#).

Contact Records – Real Time

Requires: [Amazon Data Streams/Firehose](#), [HEC Token](#) and [Lambda setup](#).

Connect Instance Metrics – CloudWatch Metrics

Requires: [Splunk Add-On for AWS](#), [CloudWatch Input in Add-on](#) and [IAM Account](#), [Enabling Contact Flow Logs](#).

Sentiment Analysis AKA Contact Lens

Requires: [SQS](#), [Splunk Add-On for AWS](#), [SQS for S3 Input in Add-on](#), [Configuring Contact Lens](#) and [IAM Account](#).

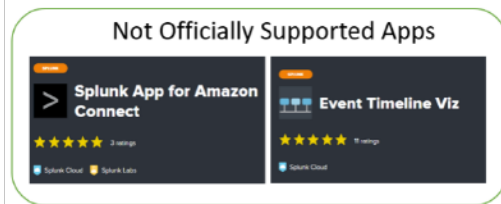
Supported vs Non-Support Apps/ Add-ons



Above are the apps and add-ons published by Splunk Inc. that are supported and maintained by Splunk. Splunk will provide customers with active support subscriptions an initial response and acknowledgement to any support request for these apps or add-ons in accordance with [Splunk Support terms](#). Splunk will also ensure compatibility of Splunk-supported apps and add-ons with future releases of applicable Software. Splunk ensures this compatibility for any Splunk-supported apps or add-ons installed in Splunk Cloud Platform before commencing Splunk Cloud Platform upgrades.

Splunk does not provide support or maintenance for apps or add-ons published by any party other than Splunk Inc., including third-party developers.

More details [here](#).



Although built by Splunk, these Apps do not carry the same support arrangement as the ones above.

An example of this is the **Splunk App for Amazon Connect**. In this case the App is designed to provide the user a collection of pre-configured dashboards and content designed to greatly reduce the initial heavy lifting in gaining visibility from your Amazon Connect Instance. Generally every user of this application will have different use cases, SLAs and metrics required with a no one size fits all.

Final notes about this guide

We hope you got some value out of this guide. This guide was a joint project by Splunk and AWS to help our users gain more value out of the platforms faster.

If you have any feedback or comments please reach out to authors as your opinion and feedback counts.

Thanks for taking the time to go through this guide.

Meet the authors



Travis Kane

Cloud Technical Strategist
at Splunk



Aurelien Plancque

Partner Solutions Architect at
Amazon Web Services (AWS)

CTA goes here xxxxx



Learn more: www.splunk.com/asksales

www.splunk.com