Securing Your Big Data Environment

Ajit Gaddam
ajit@root777.com

Abstract

Security and privacy issues are magnified by the volume, variety, and velocity of Big Data. The diversity of data sources, formats, and data flows, combined with the streaming nature of data acquisition and high volume create unique security risks.

This paper details the security challenges when organizations start moving sensitive data to a Big Data repository like Hadoop. It identifies the different threat models and the security control framework to address and mitigate security risks due to the identified threat conditions and usage models. The framework outlined in this paper is also meant to be distribution agnostic.

***Keywords**: Hadoop, Big Data, enterprise, defense, risk, Big Data Reference Framework, Security and Privacy, threat model*

1 Introduction

The term "Big Data" refers to the massive amounts of digital information that companies collect. Industry estimates on the growth rate of data is roughly double every two years, from 2500 Exabytes in 2012 to 40,000 Exabytes in 2020 [1]. Big data is not a specific technology. It is a collection of attributes and capabilities.

NIST defines Big Data as the following [2]:
Big Data consists of extensive datasets, primarily in the characteristics of volume, velocity, and/or variety that require a scalable architecture for efficient storage, manipulation, and analysis.

Securosis research [3] adds additional characteristics for a particular environment to qualify as 'Big Data'.

1. It handles a petabyte of data or more
2. It has distributed redundant data storage
3. Can leverage parallel task processing
4. Can provide data processing (MapReduce or equivalent) capabilities
5. Has extremely fast data insertion
6. Has central management and orchestration
7. Is hardware agnostic
8. Is extensible where its basic capabilities can be augmented and altered

Security and privacy issues are magnified by the volume, variety, and velocity of Big Data. The diversity of data sources, formats, and data flows, combined with the streaming nature of data acquisition and high volume create unique security risks.

It is not merely the existence of large amounts of data that is creating new security challenges for organizations. Big Data has been collected and utilized by enterprises for several decades. Software infrastructures such as Hadoop enable developers and analysts to easily leverage hundreds of computing nodes to perform data-parallel computing which was not there before. As a result, new security challenges have arisen from the coupling of Big Data with heterogeneous compositions of commodity hardware with commodity operating systems, and commodity software infrastructures for storing and computing on data. As Big Data expands at the different enterprises, traditional security mechanisms tailored to securing small-scale, static data and data flows on firewalled and semi-isolated networks are inadequate. Similarly, it is unclear how to retrofit provenance in an enterprise's existing infrastructure. Throughout this document, unless explicitly called out, Big Data will refer to the Hadoop framework and its common NoSQL variants (e.g. Cassandra, MongoDB, Couch, Riak, etc.).

This paper details the security challenges when organizations start moving sensitive data to a Big Data repository like Hadoop. It provides the different threat models and the security control framework to address and mitigate the risk due to the identified security threats. In the following sections, the paper describes in the detail the architecture of the modern Hadoop ecosystem and identify the different security weaknesses of such systems. We then identify the different threat conditions associated with them and their threat models. This paper concludes the analysis by providing a reference security framework for an enterprise Big Data environment.

2 Hadoop Security Weakness

Traditional Relational Database Management Systems (RDBMS) security has evolved over the years and with many 'eyeballs' assessing the security through various security evaluations. Unlike such solutions, Hadoop security has not undergone the same level of rigor or evaluation for that matter and thus can claim little assurance of the implemented security.

Another big challenge is that today, there is no standardization or portability of security controls between the different Open-Source Software (OSS) projects and the different Hadoop or Big Data vendors. Hadoop security is completely fragmented. This is true even when the above parties implement the same security feature for the same Hadoop component. Vendors and OSS parties' force-fit security into the Apache Hadoop framework.

2.1 Top 10 Security & Privacy Challenges

The Cloud Security Alliance Big Data Security Working Group has compiled the following as the Top 10 security and privacy challenges to overcome in Big Data [4].

1. Secure computations in distributed programming frameworks
2. Security best practices for non-relational data stores
3. Secure data storage and transactions logs
4. End-point input validation/filtering
5. Real-time security monitoring
6. Scalable privacy-preserving data mining and analytics
7. Cryptographically enforced data centric security
8. Granular access control
9. Granular audits
10. Data provenance

The above challenges were grouped into four broad components by the Cloud Security Alliance. They were:

<u>Infrastructure Security</u>
- Secure computations in distributed programming frameworks
- Security best practices for non-relational data stores

<u>Data Privacy</u>
- Scalable privacy-preserving data mining and analytics
- Cryptographically enforced data centric security
- Granular access control

<u>Data Management</u>
- Secure data storage and transactions logs
- Granular audits
- Data provenance

<u>Integrity & Reactive Security</u>
- End-point input validation/filtering
- Real-time security monitoring

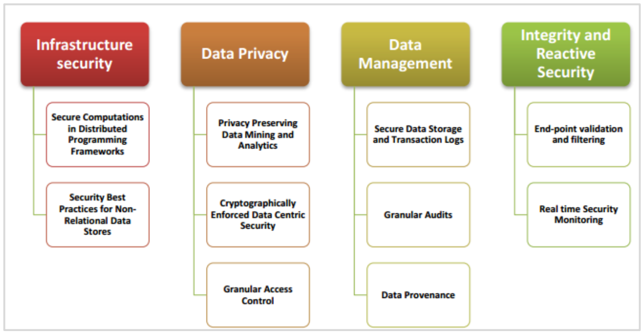

Infrastructure security
Secure Computations in Distributed Programming Frameworks
Security Best Practices for Non-Relational Data Stores
Data Privacy
Privacy Preserving Data Mining and Analytics
Cryptographically Enforced Data Centric Security
Granular Access Control
Data Management
Secure Data Storage and Transaction Logs
Granular Audits
Data Provenance
Integrity and Reactive Security
End-point validation and filtering
Real time Security Monitoring

Figure 1: CSA- classification of the Top 10 Challenges

2.2 Additional Security Weaknesses

The earlier section regarding Cloud Security Alliance list is an excellent start and this research and paper significantly adds to it. Where possible, effort has been made to map back to the categories identified in the CSA work. This section lists some additional security weaknesses associated with Open Source Software (OSS) like Apache Hadoop. It is meant to give the reader an idea of the possible attack surface. However it's not meant to be exhaustive which subsequent sections will provide and add to.

<u>Infrastructure Security & Integrity</u>
- The Common Vulnerabilities and Exposures (CVE) database only shows four reporting and fixed Hadoop vulnerabilities over the past three years. Software, even Hadoop, is far from perfect. This could either reflect that the security community is not active or that most of vulnerability remediation happens internally within the vendor environments themselves with no public reporting.
- Hadoop security configuration files are not self-contained with no validity checks prior to such policies being deployed. This usually results in data integrity and availability issues.

<u>Identity & Access Management</u>

- Role Based Access Control (RBAC) policy files and Access Control Lists (ACLs) for components like MapReduce and HBase are usually configured via clear-text files. These files are editable by privileged accounts on the system like *root* and other application accounts.

Data Privacy & Security

- All issues associated with SQL injection type of attacks don't go away. They move with Hadoop components like Hive and Impala. SQL prepare functions are currently not available which would have enabled separation of the query and data
- Lack of native cryptographic controls for sensitive data protection. Frequently, such security is provided outside the data or application stack.
- Clear-text data might be sent when communicating between DataNode to DataNode since data locality cannot be strictly enforced and the scheduler might not be able to find resources next to the data and force it to read data over the network.

3 Big Data Security Framework

The following section provides the target security architecture framework for Big Data platform security. The core components of the proposed Big Data Security Framework are the following:

1. Data Management
2. Identity & Access Management
3. Data Protection & Privacy
4. Network Security
5. Infrastructure Security & Integrity

The above '5 pillars' of Big Data Security Framework are further decomposed into 21 sub-components, each of which are critical to ensuring the security and mitigating the security risk and threat vectors to the Big Data stack. The overall security framework is shown below.

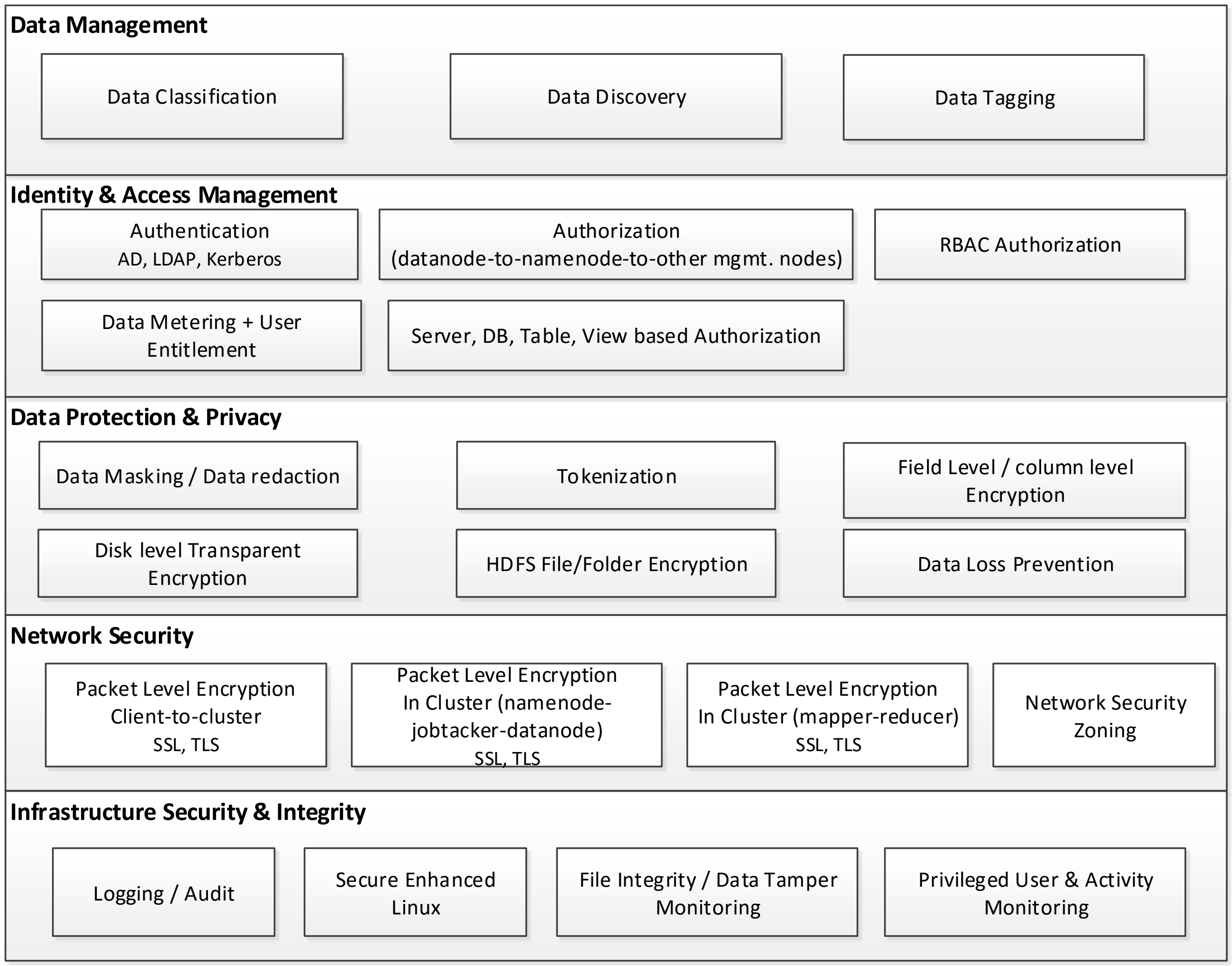

Data Management
Data Classification
Data Discovery
Data Tagging
Identity & Access Management
Authentication AD, LDAP, Kerberos
Authorization (datanode-to-namenode-to-other mgmt. nodes)
RBAC Authorization
Data Metering + User Entitlement
Server, DB, Table, View based Authorization
Data Protection & Privacy
Data Masking / Data redaction
Tokenization
Field Level / column level Encryption
Disk level Transparent Encryption
HDFS File/Folder Encryption
Data Loss Prevention
Network Security
Packet Level Encryption Client-to-cluster SSL, TLS
Packet Level Encryption In Cluster (namenode-jobtacker-datanode) SSL, TLS
Packet Level Encryption In Cluster (mapper-reducer) SSL, TLS
Network Security Zoning
Infrastructure Security & Integrity
Logging / Audit
Secure Enhanced Linux
File Integrity / Data Tamper Monitoring
Privileged User & Activity Monitoring

Figure 2: Big Data Security Framework

3.1 Data Management

Data Management component is decomposed into three core sub-components. They are Data Classification, Data Discovery, and Data Tagging.

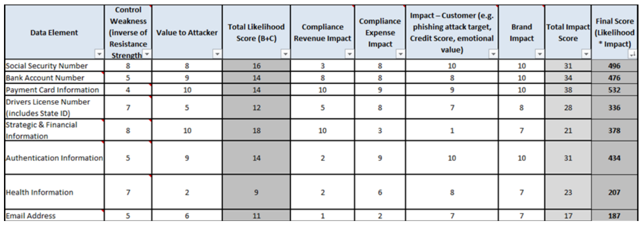
3.1.1 Data Classification

Effective data classification is probably one of the most important activities that can in-turn lead to effective security control implementation in a Big Data platform. When organizations deal with an extremely large amount of data, aka Big Data, by clearly being able to identify what data matters, what needs cryptographic protection among others, and what fields need to be prioritized first for protection, more often than not determine the success of a security initiative on this platform.

The following are the core items that have been developed over time and can lead to a successful data classification matrix of your environment.

1. Work with your legal, privacy office, Intellectual Property, Finance, and Information Security to determine all *distinct* data fields. An open bucket like health data is not sufficient. This exercise encourages the reader to go beyond the symbolic policy level exercise.
2. Perform a security control assessment exercise.
   a. Determine location of data (e.g. exposed to internet, secure data zone)
   b. Determine number of users and systems with access
   c. Determine security controls (e.g. can it be protected cryptographically)
3. Determine value of the data to the attacker
   a. Is the data easy to resell on the black market?
   b. Do you have valuable Intellectual Property (e.g. a nation state looking for nuclear reactor blueprints)
4. Determine Compliance and Revenue Impact
   a. Determine breach reporting requirements for all the *distinct* fields
   b. Does loss of a particular data field prevent you from doing business (e.g. card holder data)
   c. Estimate re-architecting cost for current systems (e.g. buying new security products)
   d. Other costs like more frequent auditing, fines and judgements and legal expenses related to compliance
5. Determine impact to the owner of the PII data (e.g. a customer)
   a. Does the field cause phishing attacks (e.g. email) vs. just replace it (e.g. loss of a credit card)

The following figure is a sample representation of certain Personally Identifiable Data fields



Data Element	Control Weakness (inverse of Resistance Strength	Value to Attacker	Total Likelihood Score (B+C)	Compliance Revenue Impact	Compliance Expense Impact	Impact – Customer (e.g. phishing attack target, Credit Score, emotional value)	Brand Impact	Total Impact Score	Final Score (Likelihood * Impact)
Social Security Number	8	8	16	3	8	10	10	31	496
Bank Account Number	5	9	14	8	8	8	10	34	476
Payment Card Information	4	10	14	10	9	9	10	38	532
Drivers License Number (includes State ID)	7	5	12	5	8	7	8	28	336
Strategic & Financial Information	8	10	18	10	3	1	7	21	378
Authentication Information	5	9	14	2	9	10	10	31	434
Health Information	7	2	9	2	6	8	7	23	207
Email Address	5	6	11	1	2	7	7	17	187

Figure 3: Data Classification Matrix

3.1.2 Data Discovery

The lack of situational awareness with respect to sensitive data could leave an organization exposed to significant risks. Identifying whether sensitive data is present in Hadoop, where it is located and subsequently triggering the appropriate data protection measures, such as data masking, data redaction, tokenization or encryption is key.

- For structured data going into Hadoop, such as relational data from databases, or, for example, comma-separated values (CSV) or JavaScript Object Notation (JSON)-formatted files, the location and classification of sensitive data may already be known. In this case, the protection of those columns or fields can occur programmatically, with, for example, a labeling engine that assigns visibility labels/cell level security to those fields.
- With unstructured data, the location, count and classification of sensitive data becomes much more difficult. Data discovery, where sensitive data can be identified and located, becomes an important first step in data protection.

The following items are crucial for an effective data discovery exercise of your Big Data environment:

1. Define and validate the data structure and schema. This is all useful prep work for data protection activities later
2. Collect metrics (e.g. volume counts, unique counts etc.). For example, if a file has 1M records but it is duplicate of a single person, it is a single record vs. 1M records. This is very useful for compliance but more importantly risk management.

3. Share this insight with your Data Science teams for them to build threat models, profiles which will be useful in data exfiltration prevention scenarios.
4. If you discover sequence files, work with your application teams to move away from this data structure. Instead leverage columnar storage formats such as Apache Parquet where possible regardless of the data processing framework, data mode, or programming language.
5. Build conditional search routines (e.g. only report on date of birth if a person's name is found or Credit Card # + CVV or CC +zip)
6. Account for usecases where once sensitive data has been cryptographically protected (e.g. encrypted or tokenized), what is the usecase for the discovery solution.

3.1.3 Data Tagging

Understand the end-to-end data flows in your Big Data environment, especially the ingress and egress methods.

1. Identify all the data ingress methods in your Big Data cluster. These would include all manual (e.g. Hadoop admins) or automated methods (e.g. ETL jobs) or those that go through some meta-layer (e.g. copy files or create + write).
2. Knowing whether data is coming in leveraging Command Line Interface or though some Java API or through Flume or Sqoop import of if it is being SSH'd in is important.
3. Similarly, follow the data out and identify all the egress components out of your Big Data environment.
4. This includes whether reporting jobs are being run through Hive queries (e.g. through ODBC/JDBC), through Pig jobs (e.g. reading files or Hive tables or HCatalog), or exporting it out via Sqoop or copying via REST API, Hue etc. will determine your control boundaries and trust zones.
5. All of the above will also help in data discovery activity and other data access management exercises (e.g. to implement RBAC, ABAC, etc.)

3.2 Identity & Access Management

POSIX-style permissions in secure HDFS are the basis for many access controls across the Hadoop stack.

3.2.1 User Entitlement + Data Metering

Provide users access to data by centrally managing access policies.

- It is important to tie policy to data and not to the access method
- Leverage Attribute based access control and protect data based on tags that move with the data through lineage; permissions decisions can leverage the user, environment (e.g. location), and data attributes.
- Perform data metering by restricting access to data once a normal threshold (as determined by access models + machine learning algorithms) is passed for a particular user/application.

3.2.2 RBAC Authorization

Deliver fine-grained authorization through Role Based Access Control (RBAC).

- Manage data access by role (and not user)
- Determine relationships between users & roles through groups. Leverage AD/LDAP group membership and enforce rules across all data access paths

3.3 Data Protection & Privacy

The majority of the Hadoop distributions and vendor add-ons package either data-at-rest encryption at a block or (whole) file level. Application level cryptographic protection (like field-level/column-level encryption, data tokenization, and data redaction/masking provide the next level of security needed.

3.3.1 Application Level Cryptography (Tokenization, field-level encryption)

While encryption at the field/element level can offer security granularity and audit tracking capabilities, it comes at the expense of requiring manual intervention to determine the fields that require encryption and where and how to enable authorized decryption.

3.3.2 Transparent Encryption (disk / HDFS layer)

Full Disk Encryption (FDE) prevents access via the storage medium. File encryption can also guard against (privileged) access at the node's operating-system level.

- In case you need to store and process sensitive or regulated data in Hadoop, data-at-rest encryption protects your organization's sensitive data and keeps at least the disks out of audit scope.

- In larger Hadoop clusters, disks often need to be removed from the cluster and replaced. Disk Level transparent encryption ensures that no human-readable residual data remains when data is removed or when disks are decommissioned.
- Full-disk encryption (FDE) can also be OS-native disk encryption, such as `dm-crypt`

3.3.3 Data Masking/ Data Redaction

Data masking or data redaction before load in the typical ETL process de-identifies personally identifiable information (PII) data before load. Therefore, no sensitive data is stored in Hadoop, keeping the Hadoop Cluster potentially out of (audit) scope.

- This may be performed in batch or real time and can be achieved with a variety of designs, including the use of static and dynamic data masking tools, as well as through data services.

3.4 Network Security

The Network Security layer is decomposed into four sub-components. They are data protection in-transit and network zoning + authorization components.

3.4.1 Data Protection In-Transit

Secure communications are required for HDFS to protect data-in-transit. There are multiple threat scenarios that in turn mandate the necessity for https and prevent information disclosure or elevation of privilege threat categories. Using the TLS protocol (which is now available in all Hadoop distributions) to authenticate and ensure privacy of communications between nodes, name servers, and applications.

- An attacker can gain unauthorized access to data by intercepting communications to Hadoop consoles.
- This could include communication between NameNodes and DataNodes that are in the clear back to the Hadoop clients and in turn can result in credentials/data to be sniffed.
- Tokens that are granted to the user post-Kerberos authentication can also be sniffed and can be used to impersonate users on the NameNode.

Following are the controls that when implemented in a Big Data cluster can ensure properties of data confidentiality.

1. Packet level encryption using TLS from the client to Hadoop cluster
2. Packet level encryption using TLS within the cluster itself. This includes using https between NameMode to Job Tracker to DataNode.
3. Packet level encryption using TLS in the cluster (e.g. mapper-reducer)
4. Use LDAP over SSL (LDAPS) rather than LDAP when communicating with the corporate enterprise directories to prevent sniffing attacks.
5. Allow your admins to configure and enable encrypted shuffle and TLS/https for HDFS, MapReduce, YARN, HBase UIs etc.

3.4.2 Network Security Zoning

The Hadoop clusters must be segmented into points of delivery (PODs) with chokepoints such as Top of Rack (ToR) switches where network Access Control Lists (ACLs) limit the allowed traffic to approved levels.

- End users must not be able to connect to the individual data nodes, but to the name nodes only.
- The Apache Knox gateway for example, provides the capability to control traffic in and out of Hadoop at the per-service-level granularity.
- A basic firewall that should allow access only to the Hadoop NameNode, or, where sufficient, to an Apache Knox gateway. Clients will never need to communicate directly with, for example, a DataNode.

3.5 Infrastructure Security & Integrity

The Infrastructure Security & Integrity layer is decomposed into four core sub-components. They are Logging/Audit, Secure Enhanced Linux (SELinux), File Integrity + Data Tamper Monitoring, and Privileged User and Activity Monitoring.

3.5.1 Logging / Audit

All system/ecosystem changes unique to Hadoop clusters need to be audited with the audit logs being protected. Examples include:

- Addition/deletion of data and management nodes
- Changes in management node states including job tracker nodes, name nodes
- Pre-shared secrets or certificates that are rolled out when the initial package of the Hadoop

distribution or of the security solution is pushed to the node prevent the addition of unauthorized cluster nodes.

When data is not limited to one of the core Hadoop components, Hadoop data security ends up having many moving parts and high percentage of fragmentation. Consequently, there results a sprawl of metadata and audit logs across all fragments.

In a typical enterprise, the DBAs are typically leveraged to place the security responsibility at the table, row, column, or cell level and while the configuration of file systems and system administrators, and the Security Access Control team is usually accountable for the more granular file level permissions.

Yet, in Hadoop, POSIX-style HDFS permissions are frequently important for data security or are at times the only means to enforce data security at all. This leads to questions concerning the manageability of Hadoop security. Technologies recommendations to address data fragmentation:

- **Apache Falcon** is an incubating Apache OSS project that focuses on data management. It provides graphical data lineage and actively controls the data life cycle. Metadata is retrieved and mashed up from wherever the Hadoop application stores it.
- **Cloudera Navigator** is a proprietary tool and GUI that is part of Cloudera's Distribution Including Apache Hadoop (CDH) distribution. CDH Navigator is a tool to address log sprawl, lineage and some aspects of data discovery. Metadata is retrieved and mashed up from wherever the Hadoop application stores it.
- **Zettaset Orchestrator** is a product for harnessing the overall fragmentation of Hadoop security with a proprietary combined GUI and workflow. Zettaset has its own metadata repository where metadata from all Hadoop components is collected and stored.

3.5.2 Secure Enhanced Linux (SELinux)

SELinux was created by the United States National Security Agency (NSA) as a set of patches to the Linux Kernel using Linux Security Modules (LSM). It was eventually released by the NSA under the GPL license and has been adopted by the upstream Linux kernel.

SELinux is an example of a Mandatory Access Control (MAC) for Linux. Historically Hadoop and other Big Data platforms built on top of Linux and UNIX systems have had discretionary access control. What this means for example is that a privileged user like root is omnipotent.

- By enforcing and configuring SELinux on your Big Data environment, through MAC, there is policy which is administratively set and fixed.
- Even if a user changes any settings on their home directory, the policy prevents another user or process from accessing it.
- A sample policy for example that can be implemented is to make library files executable but not writable or vice-versa. Jobs can write to /tmp location but not be able to execute anything in there. This is a great way to prevent command injection attacks among others.
- With policies configured, even if someone who is a sysadmin or a malicious user is able to gain access to root using SSH or some other attack vector, they may be able to read and write a lot of stuff. However, they won't be able to execute anything incl. potentially any data exfiltration methods.

The general recommendation is to run SELinux is permissive mode with regular workloads on your cluster, reflecting typical usage, including using any tools. The warnings generated can then be used to define the SELinux policy which after tuning can be deployed in a 'targeted enforcement' mode.

4 Final Recommendations

The following are some key recommendations in helping mitigate the security risks and threats identified in the Big Data ecosystem.

1. Select products and vendors that have proven experience in similar-scale deployments. Request vendor references for large deployments (that is, similar in size to your organization) that have been running the security controls under consideration for your project for at least one year
2. Key pillars are: Accountability, balancing network centric, access-control centric, and data centric security is absolutely critical in achieving a good overall trustworthy security posture.

3. Data-centric security, such as label security or cell-level security for sensitive data is preferred. Label security and cell-level security are integrated into the data or into the application code rather than adding data security after the fact
4. Externalize data security when possible and use data redaction, data masking or tokenization at the time of ingestion, or use data services with granular controls to access Hadoop
5. Harness the log and audit sprawl with data management tools, such as OSS Apache Falcon, Cloudera Navigator or the Zettaset Orchestrator. This helps achieve data provenance in the long run

5 Related Work

A lot of publications have been released in the recent past around Hadoop. However, there are very few to none around Big Data and Hadoop security. This is getting remediated with the book *Hadoop in Action, Second Edition* from Manning Publications [5] set to be published towards the end of 2015.

This book will integrate security as part of the overall Hadoop ecosystem including greater depth and articulation of the concepts presented in this paper.

6 Conclusion

Hadoop and big data are no longer buzz words in large enterprises. Whether for the correct reasons or not, enterprise data warehouses are moving to Hadoop and along with it come petabytes of data.

In this paper we have laid the groundwork for conducting future security assessments on the Big Data ecosystem and securing it. This is to ensure that Big Data in Hadoop does not become a big problem or a big target. Vendors pitch their technologies as the magical silver bullet. However, there are many challenges when it comes to deploying security controls in your Big Data environment.

This paper also provides the Big Data threat model which the reader can further expand and customize to their organizational environment. It also provides a target reference architecture around Big Data security and covers the entire control stack.

Hadoop and big data represent a green field opportunity for security practitioners. It provides a chance to get ahead of the curve, test and deploy your tools, processes, patterns, and techniques before big data becomes a big problem.

References

[1] EMC Big Data 2020 Projects http://www.emc.com/leadership/digital-universe/iview/big-data-2020.htm

[2] NIST Special Publication 1500-1 *NIST Big Data Interoperability Framework: Volume 1, Definitions* http://bigdatawg.nist.gov/_uploadfiles/M0392_v1_3022325181.pdf

[3] Securosis – Securing Big Data Security issues with Hadoop environments https://securosis.com/blog/securing-big-data-security-issues-with-hadoop-environments

[4] Top 10 Big Data Security and Privacy Challenges, Cloud Security Alliance, 2012 https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Big_Data_Top_Ten_v1.pdf

[5] Hadoop in Action, Second Edition by Manning Publications. ISBN: 9781617291227 http://www.manning.com/lam2/