SAPIENZA
UNIVERSITÀ DI ROMA

Titolo della tesi
Hadoop Security Evaluation

Facoltà di
Corso di laurea in
Masters in Computer Science

Candidato
Kasi Viswanath Satti
1722202

Relatore
Francesco Parisi-Presicce

Academic Year 2017/2018

Abstract

There are different ways to store and process large amount of data. Hadoop is widely used, one of the most popular platforms to store huge amount of data and process them in parallel. While storing sensitive data, security plays a key role for Data management especially when working with large amount of data, organizations faces risk if they don't take care of security of their data. Security was not that much considered when Hadoop was initially designed. Hadoop built-in security is inconsistent, the commercial Hadoop distributions from software vendors like Hortonworks have additional, proprietary security that is not included in the Hadoop. Data needs to be protected in order to avoid data security breaches. In this Thesis the security of the traditional Hadoop is evaluated based on the five key principles Authentication, Authorization, Data Protection, Protecting the Network, Auditing, by considering Hadoop Distributions and suggests a security flow in order to protect the data in the cluster.

ACKNOWLEDGEMENT

I would like to sincerely thank my supervisor **Francesco Parisi-Presicce.** For his great support in guiding me through the right track by setting weekly meetings while doing thesis. Francesco helped me a lot with his patience through security analytical materials and also in structuring the thesis report. He was really helpful in providing me with a holistic view of security concepts and also helped me to have more technical view of the security tools.

I am grateful to my friends and family who supported me through the thesis work.

CONTENTS

1. Big Data and Cloud Computing

1.1 Big Data:

Big Data refers to data that is so large in volume or complexity that current technology is not able to store and process it efficiently. Such a requirement has led to the advent of modern software like Apache Hadoop which uses Map Reduce to process and analyse large volumes of data by parallelizing the processing and using distributed hardware. Every day 2.5 exabytes (2.5×10^{18}) of data are generated. By 2025 International Data Corporation (IDC) predicts that there will be 163 zettabytes of data. [1]

1.2 Big Data Characteristics:

Volume

The quantity of generated and stored data.

Variety

The type and nature of the data. This helps people who analyse it to effectively use the result insight.

Velocity

In this context, the speed at which the data is generated and processed to meet the demands and challenges that lie in the path of growth and development.

Veracity

The data quality of captured data can vary greatly, affecting the accurate analysis.

1.3 Big Data Challenges:

Data Storage

Data is generated from various sources like Web, social media, Internet of Things etc. Data may be structured when fetched from relational data stores, unstructured when fetched from sources like social media or semi-structured depending on the source. All data go into the data store, storing all the data generated is the biggest challenge. [2]

Data Management

Data can be distributed geographically, managed by multiple entities. Data may be in different formats need to be incorporated at all levels of satisfactorily manage the data. This is difficult due to huge volume and variety. High velocity of data intake makes it difficult to validate and process in real-time. [2]

Risk Detection:

Risks are associated with breaches and leaks from confidential data. Protecting confidential data is a basic requirement when handling sensitive data. Breaches and leaks can lead to loss of business if user confidential data is revealed to outside world. Hence risk detection has to be a proactive process which means breaches and leaks should be detected and prevented before they have a potential to happen. [2]

Data Security and Privacy

Data Today is undergoing increased adoption of cloud technologies due to the ability of buying processing power and storage on-demand. This is exposing enterprise data to outside world and hence novel big data security measures need to be taken to secure data from falling into wrong hands. [2]

1.4 Cloud Computing:

National Institute of Standards and Technology (NIST) defines cloud computing as follows:

A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (eg: networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud computing is a technology which depends on sharing of computing resources than having local servers or personal devices to handle the applications. In cloud computing the word cloud means the internet so cloud computing means a type of computing in which services are delivered through the internet. The goal of cloud computing is to make use of increasing computing power to execute millions of instructions per second. Cloud computing uses networks of a large group of servers with specialized connection to distribute data processing among the servers. Instead of installing a software suite for each computer, this technology requires to install a single software in each computer that allows users to log into a web-based service and which also hosts all the programs required by the user. [3]

1.5 Top Threats for Cloud:

For cloud risk assessment, the CSA (Cloud Security Alliances). CSA conducted a survey among the experts and stakeholders to gain an insight into their perception on the threats against the cloud and published the results in a document titled 'The Treacherous 12' cloud computing top threats in 2016-17.

1.Data Breach

A data breach is an incident in which sensitive, protected or confidential information is released, viewed, stolen or used by an individual who is not authorized to do so. A data breach may be the primary objective of a targeted attack or may simply be the result of human error, application vulnerabilities or poor security practices. [4]

2.Insufficient Identity, credential and Access Management

Data breaches and enabling of attacks can occur because of a lack of scalable identity access management system, failure to use multifactor Authentication, weak password use, and a lack of ongoing automated rotation of cryptographic keys, passwords and certificates. Credentials and cryptographic keys must not be embedded in source code or distributed, because there is a significant chance of discovery and misuse. Keys

need to be appropriately secured and a well-secured Public Key Infrastructure (PKI) is needed to ensure key-management activities are carried out. [4]

3.Insecure Interfaces and API's

Cloud computing providers expose a set of software user interfaces (UIs) or Application Programming Interfaces (APIs) that customers use to manage and interact with cloud services. Provisioning, management, monitoring, all performed with these interfaces. The security and availability of general cloud services dependent on the security of these basic APIs. From Authentication and access control to encryption and activity monitoring, these interfaces must be designed to protect against both accidental and malicious attempts to circumvent policy. [4]

4.System Vulnerabilities

System Vulnerabilities are exploitable bugs in programs that attackers can use to intrude a computer system for the purpose of stealing data, taking control of the system or disrupting service operations. [4]

5.Account Hijacking

Attack methods such as phishing, fraud exploitation of software vulnerabilities still achieve results. If an attacker gains access to the credentials, they can eavesdrop on our activities and transactions, manipulate data, return falsified information and redirect our clients to illegitimate sites. Account or service instances may become a new base for attackers, from here they launch subsequent attacks. [4]

6.Malicious Insiders

A Malicious Insider threat to an organization is a current or former employee, contractor, or other business partner who has or had authorized access to an organization's network, system or data and intentionally exceeded or misused that access in a manner that negatively affected the confidentiality, integrity, or availability of the organization's information. [4]

7.Advanced Persistent Threats

Advanced Persistent Threat (APT) is a set of cautious and continuous computer hacking processes often managed by a person targeting a specific entity. APTs are a form of cyberattack that gain access to systems to establish a foothold in the computing infrastructure of target companies from which they smuggle data and intellectual property. [4]

8.Data Loss

Data stored in the cloud can be lost for reasons other than malicious attacks. An accidental deletion by cloud service provider, a physical disaster such as fire or earthquake can lead to permanent loss of customer data, if a customer encrypts his data before uploading it to the cloud but loses the encryption key, the data will be lost, cloud consumers has to take adequate measures to back up data. [4]

9.Insufficient Due Diligence

Reasonable steps taken by a person to avoid offence, when executives creates business strategies, cloud technologies and Cloud Service Providers must be considered. Developing a good roadmap and checklist for due diligence when evaluating technologies and cloud service providers is essential for the greatest chance of success. [4]

10.Abuse and Nefarious use of Cloud Services

Poorly secured cloud service deployments, free cloud service trials and fraudulent account sign-ups via payment instrument fraud expose cloud computing models such as IaaS, PaaS, and SaaS to malicious attacks. Malicious actors may leverage cloud computing resources to target users, organizations or other cloud providers. Examples of misuse of cloud service-based resources include launching DDOS attacks, email spam and phishing, mining for digital currency, large-scale automated click fraud, brute-force compute attacks of stolen credential databases and hosting of malicious or pirated content. [4]

11.Denial of Service

Denial-of-Service (DOS) attacks are attacks meant to prevent users of a service from being able to access their data or their applications. By forcing the targeted cloud service to consume inordinate amounts of finite system resources such as processor power, memory, disk space or network bandwidth, the attacker, as is the case in distributed denial-of-service (DDOS) attacks causes an intolerable system slowdown and leaves all legitimate service users confused as to why the service is not responding. [4]

12.Shared Technology Vulnerabilities

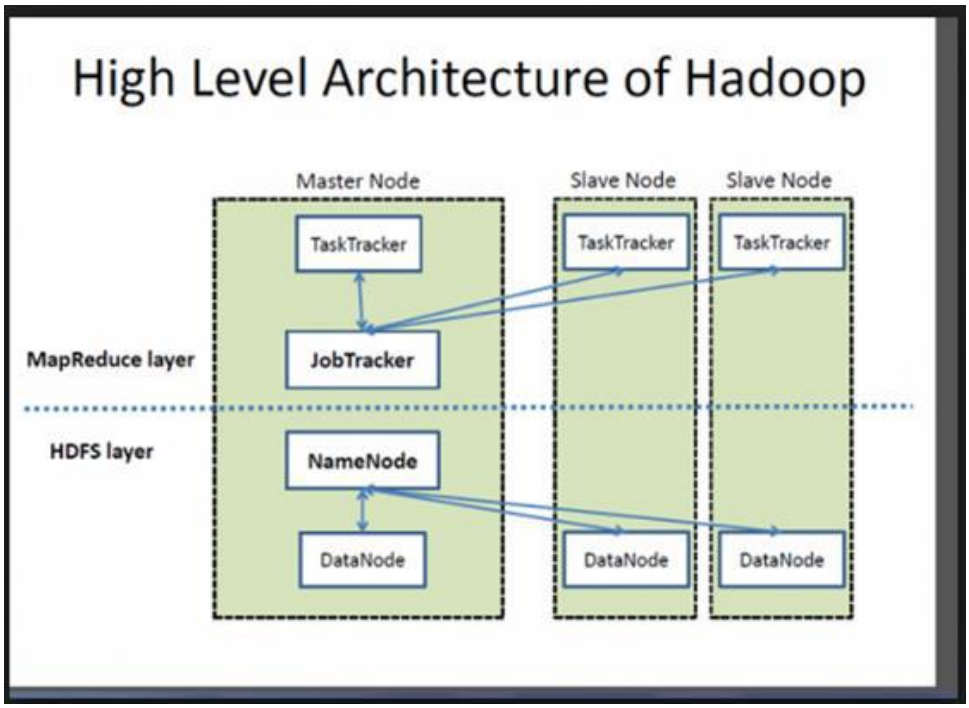
Cloud service providers deliver their services scalable by sharing infrastructure. Cloud technology divides the "as a service" offering without substantially changing the hardware/software sometimes at the expense of security. Underlying components such as CPU caches, GPUs etc that comprise the infrastructure supporting cloud services deployment may not have been designed to offer storing isolation properties for a multitenant architecture (IaaS), re-deployable platform (Paas) or multi-customer applications (SaaS).

This can lead to shared technology vulnerabilities that can potentially be exploited in all delivery models. A defence in-depth strategy is recommended and should include compute, storage, network, application and user security enforcement and monitoring, whether the service model is IaaS, PaaS, or SaaS. The key is that a single vulnerability can lead to compromise across an entire provider's cloud. [4]

2. Hadoop

2.1 What is Hadoop:

Hadoop is an open source framework to store and process large amounts of data. Hadoop has been developed under an Apache License. It is implemented to scale from a single cluster to thousands of servers.



Hadoop Architecture

Hadoop consists of two main modules: Hadoop Distributed File Systems(HDFS), Map Reduce

Hadoop Distributed File Systems:

Hadoop can work directly with any distributed file system such as Local File System, S3 etc, but the most common file system used by Hadoop is the Hadoop Distributed File System(HDFS).

The Hadoop Distributed File System is based on the Google File System(GFS) that is designed to run on large clusters of machines in a reliable, fault-tolerant manner.

HDFS uses a master/slave architecture where master consists of a single Name Node that manages the file system metadata and one or more slave Data Nodes that store the actual data. [5]

Name Node:

Name Node keeps the directory tree of all the files in the file system in the form of metadata. It does not store the data of these files itself. Client applications talk to Name Node when they want to add/copy/ move/ delete a file. Name Node responds by returning a list of Relevant Data Node servers where the data lives. [6]

Data Node:

Data node stores the actual data in the form of blocks. A file system has more than one data node, with data replicated across them. [7]

Map Reduce:

Map Reduce is a framework and programming model for Distributed Computing, using which we can write applications to process huge amount of data in parallel.

The Map Reduce framework consists of a single master Job Tracker and one slave Task Tracker per cluster-node.

Job Tracker:

Job Tracker responsible for resource Management, scheduling the jobs on the slaves, monitoring them and re-executing the failed tasks.

Task Tracker:

The Task Tracker execute the tasks as directed by the master and provide task-status information to the master periodically.

The Map Reduce programming model contains two important tasks, namely Map task and Reduce task.

Map task takes set of data and converts it into another set of data, where individual elements are broken down into key/value pair tuples.

Reduce task takes the output from Map task as an input and combines those data tuples into a smaller set of tuples. Reduce task always performs after Map task. [5]

2.2 How Hadoop works

When storing and retrieving the files from HDFS, Name Node (Master) and Data Node (slave) are considered, if a user wants to store the file into HDFS, initially user sends request to Name Node, Name Node is the master node it maintains meta data (Data Node locations, amount of space available in Data Node). Name Node returns the Data node locations to the user to store data. Now user knows the locations of Data Node, the user sends request to Data Node, Data Node stores the files in the form of Blocks. This information is stored into Name Node meta data. For retrieving the data similar process takes place.

When processing the data which is in HDFS using Map Reduce, Job Tracker (Master) and Task Tracker (slave) are considered, if a user wants to analyse the data which is in HDFS using Map Reduce, the user writes Map Reduce code and sends the Request to Job Tracker, The Job Tracker is the Master node can talk With Name Node to get the location of data (i.e Data Node locations) where the files are stored. After getting the locations Job Tracker schedules the task to the Task Tracker and monitors each task, where each task in different Data Nodes runs in parallel.

2.3 Advantages and Disadvantages of Hadoop: [8]

Advantages of Hadoop:

1.Scalable

Hadoop is highly scalable storage platform, because it can store's and distribute very large data sets across hundreds of inexpensive servers that operate in parallel. Hadoop enables businesses to run applications on thousands of nodes involving many thousands of terabytes of data.

2.Cost effective

Hadoop offers a cost effective storage solution for businesses exploding data sets. The problem with traditional Relational Database Management systems is that it extremely cost prohibitive to scale to such a degree in order to process massive amount of data.

3. flexible

Hadoop enables businesses to easily access new data sources and tap into different types of data both structured and unstructured.

4. Fault Tolerant

A key advantage of using Hadoop is its fault tolerance. When data is sent to an individual node, that data is also replicated to other nodes in the cluster, which means that in the event of failure, there is another copy available for use.

Disadvantages of Hadoop:

1.Security Concerns

Hadoop security is disabled by default due to sheer complexity, so data could be at huge risk.

2.Vulnerable by Nature

The framework is written almost entirely in java, one of the most widely used yet controversial programming languages in existence. Java has been heavily exploited by cybercriminals and as a result, implicated in numerous security breaches.

3. Not Fit for Small Data

Due to its high capacity design, the Hadoop Distributed File System, lacks the ability to efficiently support the random reading of small files. As a result, it is not recommended for organizations with small quantities of data.

3. Hadoop Security

Hadoop become a more popular platform to store and process large amount of data. When Hadoop was originally designed, security was not considered. Initially users and services in Hadoop were not authenticated, all users or programs had the same level of access to the data in the clusters .Any one can enter into the cluster as a valid user and can read the data , there is no trust worthiness of data , data is not encrypted

Threats Identified [9]

- An unauthorized user may access an HDFS file via RPC or HTTP protocols and could execute random code or carry out further attacks.
- An unauthorized client may read/write a data block of a file at a Data Node via the pipeline streaming data-transfer protocol
- An unauthorized user may eavesdrop/sniff to data packets being sent by Data Nodes to client.
- An unauthorized user may access intermediate data of Map job via its Task Trackers HTTP shuffle protocol.
- An attacker who can enter the data center either physically or electronically can steal the data they want, since the data is un-encrypted and there is no authentication enforced for access.
- Hadoop security is not properly addressed by firewalls, once a firewall is breached; the cluster is wide-open for attack. Firewalls offer no protection for data-at-rest or data-in-motion.

Hadoop became a more popular platform to store and process large amount of data, Security should be consider while storing and processing this large amount of sensitive data, to avoid security breaches.

3.1 Core Pillars of Information Security: [10]

Information security has relied upon the following pillars:

- Confidentiality – only allow access data for which the user is permitted
- Integrity – ensure data is not tampered or altered by unauthorized users
- Availability – ensure systems and data are available to authorized users when they need it

Considering each pillar in turn will assist in producing a robust security control.

To secure the Hadoop platform, consider the five key principles [11]

- Authentication (user is who he claims to be)
- Authorization (Manage Access to Resource)
- Data Protection (protect against leakage of data)
- Protecting the network (secure Hadoop cluster's network)
- Auditing (ensure compliance through audit trail)

3.2 Authentication:

If Hadoop is configured with all of its defaults, Hadoop does not do any authentication of users, which means that no attempts made to verify the identity of users who interact with the cluster.

In an insecure cluster Name Node and Job Tracker do not have any authentication. If user make a request, and say you are HDFS or Map Reduce, the Name Node or Job Tracker will both believe that and allows to do whatever the HDFS or Map Reduce users have the ability to do. The Hadoop cluster is insecure, to make it secure cluster services need to authenticate callers, which means some information must be passed with remote calls to declare a caller's identity and authenticate that identity.

Hadoop can use the Kerberos protocol to ensure that when someone makes a request, they really are who they say they are, Hadoop daemons use Kerberos to perform authentication services running with in the Hadoop cluster itself so that only authenticated HDFS Data Nodes can join HDFS file system. [12]

3.2.1 Why do we need Kerberos for Authentication:

Kerberos is a computer network authentication protocol that works on the basis of tickets to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner. [26] it was written to support centrally managed accounts in a local area network, one in which administrators manage individual accounts. This is actually much simpler to manage than Public Key Infrastructure (PKI)-certificate based system.

Rather than building in elaborate authentication protocols at each server, Kerberos provides a centralized Authentication Server (AS) whose function is to authenticate users to servers and servers to users.

Kerberos when running with Hadoop cluster, users of Hadoop do not need to worry about the implementation details, developers of core Hadoop writing the code to interact with a Hadoop cluster and applications running in it do need to know those details. [12]

3.2.2 What is Kerberos:

Kerberos is an authentication mechanism developed at Massachusetts Institute of Technology (MIT).

It is designed to provide strong authentication for client/server applications by using secret key cryptography.

Key Distribution Center (KDC)

The KDC is comprised of three components

- Kerberos Database
- Authentication Service (AS)
- Ticket-Granting Service (TGS)

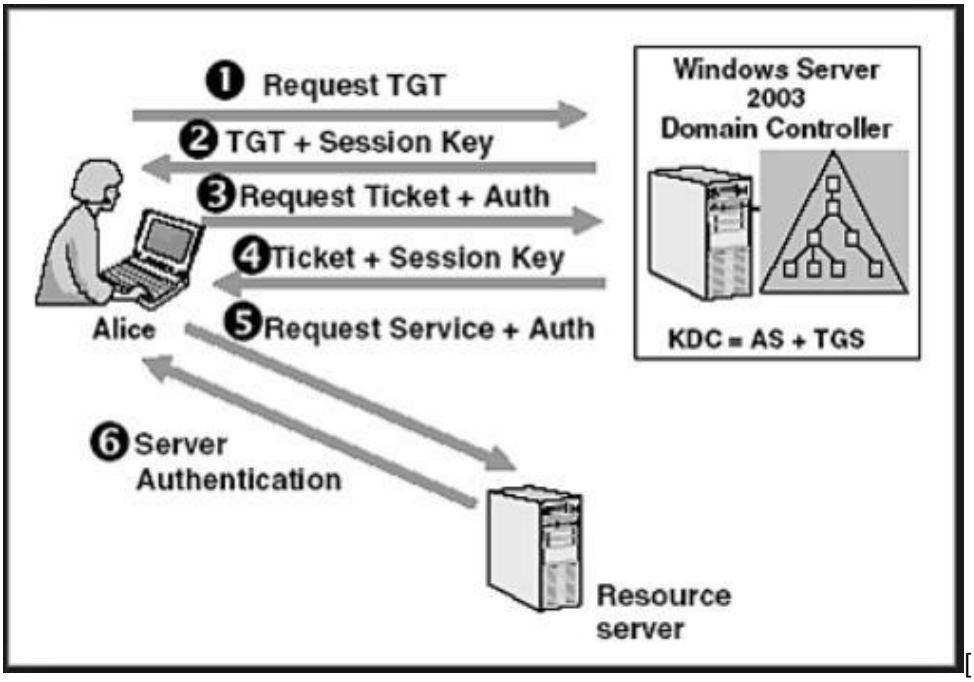
A Kerberos Database contains all of Kerberos principals, their passwords and other administrative information.

Kerberos provides centralized Authentication Server (AS) to authenticate users to servers and servers to users. A client who wishes to access a server will first Authenticate itself with the Authentication Server. The Authentication Server shares a unique secret key with each server. The Authentication server will return an authentication ticket called a Ticket Granting Ticket (TGT).

Kerberos also introduces the concept of a Ticket-Granting Server (TGS). A client that wishes to use a service has to receive a ticket – a time limited cryptographic message giving it access to the server.

The two servers combined make up a Kerberos Key Distribution Center.

Kerberos Work Flow



[13]

Kerberos Work Flow

1. User logs on to work station and request service on host, the work station sends a message to Authentication Server requesting a Ticket-granting Ticket(TGT).
2. Authentication Server Verifies user's access right in database, creates Ticket Granting Ticket and Session Key. Results are encrypted using key derived from user's password, the encrypted results sent back to the user work station
3. Work Station prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator to TGS. Authenticator contains user's name, network address, time to TGS. Here the user proves his identity by sending the ticket, authenticator encrypted with session key received in step 2.
4. Ticket Granting Server decrypts ticket and authenticator verifies request, then creates Ticket for Requested Server. The ticket Granting Server returns the ticket to the user Work Station. The return message contains two copies of server's session key one encrypted with client password, one encrypted by server's password.
5. Work Station send ticket received in step 4 and authenticator to server which is Service Request to Server
6. Server Verifies that ticket and authenticator match, then grants access to service.

Advantages of Kerberos :

1. A secure Communicaton channel is not needed for authentication, because the password is never transmitted from one party to another.
2. Kerberos is stable and widely supported on all platforms.

Disadvantages of Kerberos :

1. The authentication server is a single point of failure.
2. Kerberos has strict time requirements, and the clocks of the involved hosts must be synchronized with in configured limits.
3. As we are work with cluster of computers there is a chance of excess load to Key Distribution Center, this can over load KDC. [14]

3.2.3 Role of Kerberos in Hadoop:

The Kerberos mode in Hadoop eco system provides a secure Hadoop environment, the Kerberos service offers strong user Authentication, so in Kerberos mode only authorized users can access services, thereby preventing an unauthorized access to services. The client and the Hadoop Services (Name Node, Data Node, Job Tracker, Task Tracker) authenticate each other using Kerberos. The user, Name Node, Data Node are Known to KDC.

First the user Request the KDC for a ticket and the Kerberos server returns an encrypted Ticket (TGT). The TGT is decrypted and presented again to the authentication service of the Kerberos server requesting for a service ticket. The returned service ticket is used to access the Name Node service. [15]

To complement Kerberos, other mechanism such as Delegation Token, Block Access Token, Job Token are added.

Delegation Token

Hadoop Relies on Kerberos a three-party Authentication protocol, it has a problem when using with Hadoop. In Hadoop as files are distributed and stored at different server, imagine if we want to store a file it has to store into different worker nodes as scheduled by Job Tracker, and if all the worker nodes have to Authenticate via Kerberos using Ticket Granting Ticket, thousands of these tasks try to Authenticate

themselves at around the same time, The Kerberos Key Distribution Center would become the bottleneck. [15], thus Delegation Tokens were introduced as a light-weight Authentication method to complement Kerberos Authentication.

Delegation Token Authentication is a two-party protocol based on SASL digest-MD5 and works as follows

- The client initially Authenticates with each server via Kerberos and obtains a Delegation Token from that server.
- The Delegation Token is issued by the Name Node upon a client request and has the same semantics for expiration and renewal as a TGT granted by a Kerberos KDC.
- The client uses the Delegation Token for subsequent Authentications with the server instead of using Kerberos. [28]

User Authenticates to the Name Node using Kerberos and gets delegation Token from Name Node, as part of the Job Submission user passes the Delegation Token to Job Tracker, Job Tracker copies the Delegation token to all Task Trackers

Apache Hadoop RPC client now can talk securely with server either using tokens or Kerberos, if a token exists for a service tokens are used for secure communication, if a token doesn't exist the Kerberos is used.

Format of Delegation Token: [9]

Token ID = owner, renewer, issue date, max date, sequence number

Token Authenticator = HMAC-SHA1(Master Key, Token ID)

Delegation Token = Token ID, Token Authenticator

owner: The user who owns the token

renewer: The user who can renew the Token

issue date: Epoc Time when the token was issued

max date: Epoc time when the token can be renewed until

sequence number: UUID to identify the token

Master Key: Chosen by Name Node Randomly, Name Node use this master key to create delegation token. Master Key is stored in Name Node

Block Access Token

The Data Node typically stores lots of data blocks, and they could belong to different files owned by different users. When a request for reading or writing a data block comes from a user, the Data Node needs to ensure that the user is authorized to read or write the block. Hadoop defines tokens for authorizing such accesses. This token is called **Block Access Token.** Name Node generates the Block Access Token when a client makes a request for accessing a file's block for reading or for writing. [15]

Format of Block Access Token: [9]

Token ID = expiration Date, key ID, owner, block ID, access Modes

Token Authenticator = HMAC-SHA1(key, Token ID)

Block Access Token = Token ID, Token Authenticator

expiration Date: The date of expiry

Owner: The user who owns the token

Key Id in Token ID is used to identify the key which is used to generate the Block Access Token.

Block ID: block identification authenticated by a token

access Modes could be READ, WRITE, COPY, REPLACE

Block access tokens are generated by the Name Node. Name Node shares a symmetric-key (key used in generating Token Authenticator) with all the Data Nodes. Name Node computes the keyed hash of the Token ID (by using the shared secret) called Token Authenticator.

The Block access token consisting of both Token ID and Token Authenticator will be sent to Data Node. Data Node will re-compute the Token Authenticator using Token ID and the key which it shares with Name Node. If the calculated Token Authenticator is the same as Token Authenticator included in the Block Access Token, then the token is considered as valid. [9]

Job Token

Job Token is created by Job Tracker to provide Authentication between Map/Reduce tasks and Task Trackers. This token is used when task reports its status to Task Tracker. Another usage of Job Token is when Reducer task wants to fetch the map output from the Task Tracker. After Map Task finishes, map output is given to the Task Tracker, then each Reduce Task in that job contacts Task Tracker to fetch the map output. Job Token is used by Reduce Task to Authenticate itself to the Task Tracker.

Format of Job Token: [9]

Job Token Id = Job ID

Token Authenticator = HMAC-SHA1 (key, Job Token ID)

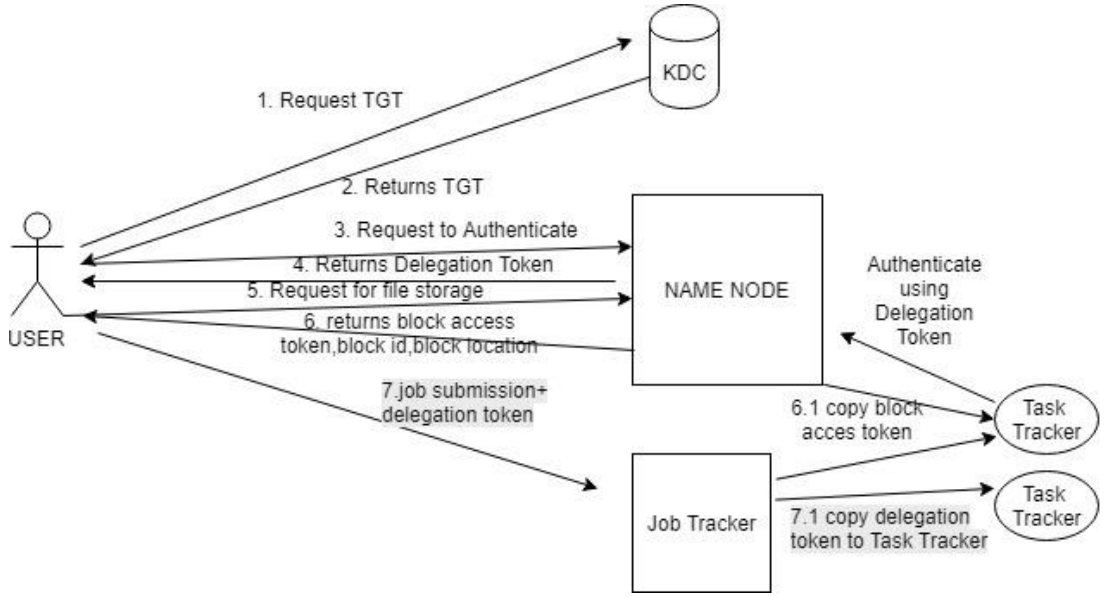
Job Token = Job Token ID, Token Authenticator

When job is submitted, Job Tracker will generate the Job Token. Job (Token Authenticator) will be stored as part of the job credential in Job Tracker directory in HDFS.

Task Trackers will read the Job Token from HDFS directory and will write the Job Token in the local disk of job directory, this directory is only visible to the one who submitted the job. Map/Reduce task will also read the Job Token from the Local directory.

Map/Reduce task sends HMAC-SHA1 (URL + current time + token Authenticator) to the Task Tracker to be Authenticated to the Task Tracker. Task Tracker computes the HMAC-SHA1 and will compare it with the HMAC-SHA1 which was sent in request. If the two hash values are same, then it proves that Map Reduce has got the same Token Authenticator which is shared with Task Tracker; After that, task Tracker and Job Tracker start Authenticating each other using HMAC-SHA1 cryptographic hash function with the Token Authenticator as the shared secret. [9]

Writing Files into HDFS:

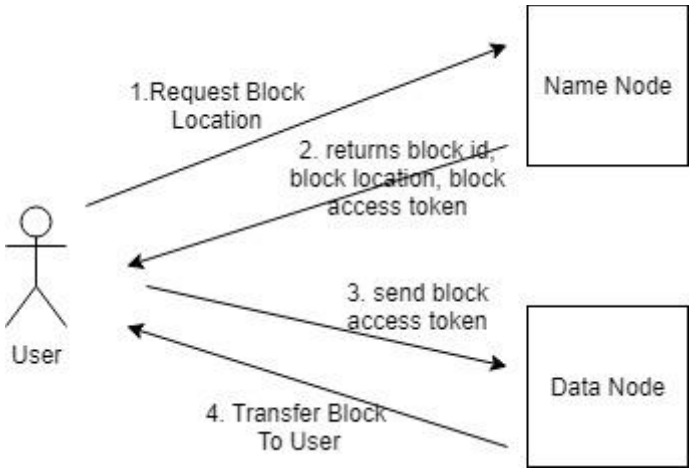


Writing files to HDFS

The different steps in writing the file:

1. User Request Ticket Granting Ticket (TGT) to Key Distribution Center.
2. Key Distribution Center returns TGT to client.
3. Using Kerberos credentials user Request Name Node to Authenticate.
4. After successful Authentication, Name Node returns Delegation Token to client, Delegation Token has the same semantics for expiration and renewal as a TGT granted by a Kerberos KDC. Name Node persists Delegation token to its meta data
5. Client Request Name Node to store the file.
6. Name Node checks the meta data and sends Block Access Token, Block Id, Block Location to user. Block Access Tokens copies to Data Nodes.
7. Now user knows Data Node location, user submit the job to Job Tracker along with Delegation Token. Job Tracker copies the Delegation Token to Task Tracker which is used for subsequent Authentication. Each Data Node verifies the block access token and then writes the data into blocks.

Reading Files from HDFS:



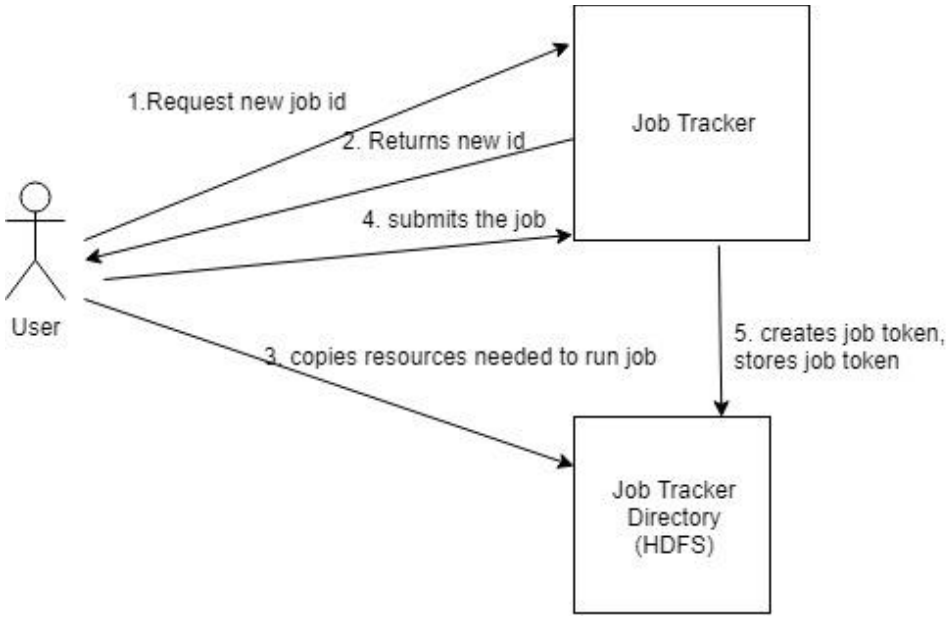
Reading Files from HDFS

The different steps in reading the file

First user logs into network and will get the Kerberos TGT, After Authentication Name Node sends delegation token to user.

1. User contacts the Name Node to request block location.
2. Name Node returns block ID's, block locations, and block access tokens to user.
3. Client will send block access token to the Data Nodes where the blocks are allocated.
4. Each Data Node verifies block access Token and it transfers block to the user.

Submitting Job to Map Reduce:

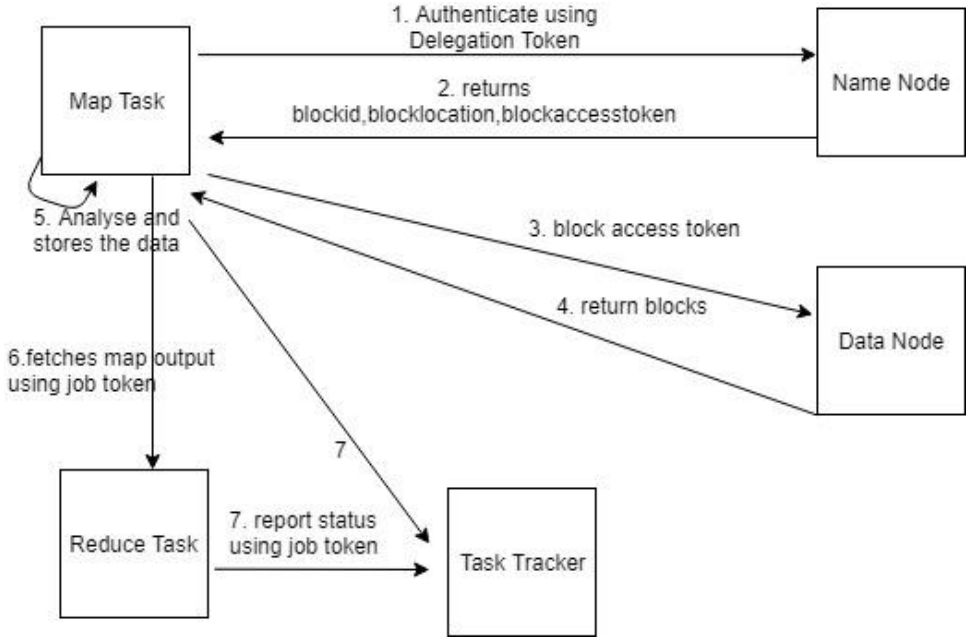


Submitting job to Map Reduce

First the user logs in to the network, is authenticated by Kerberos, and receives the Kerberos TGT. user submits a job to analyse the data in the file(s)

1. Client contacts Job Tracker via RPC, requesting a new job id
2. Job Tracker returns the new job id to the client.
3. Client copies the resources needed to run the job to Job Tracker's system directory in HDFS. In its first RPC call to Name Node, Client gets a delegation token in response and copies it to the same directory.
4. Client then submits the job to the Job Tracker and informs the Job Tracker that job is ready to execute.
5. Job Tracker creates a job Token.
6. Job Token is used to identify tasks to the framework and is stored together with job resources in its system directory in HDFS. Job Tracker initializes the job and creates Map/Reduce tasks for it. [9]

Map/Reduce Task Execution:



Map Reduce Task Execution

Task Tracker sends heartbeat to the Job Tracker to receive task. Task Tracker receives the task and makes a local directory for the task, copies job resources which includes Job Token and delegation token to its local directory.

1. Map Task contacts the Name Node through RPC and Authenticates itself to Name Node with the delegation token.
2. Name Node checks the file permission, if it is ok will return block ID's, block locations and block access token to the Map Tasks.
3. Map Task sends block access token to Data Node.
4. Data Node verifies the token, if ok transfers the block to Map Task.
5. Map Task Analyses the data and stores the map output in the local directory/
6. Reducer Task fetches the map output from Map Task via HTTP, using Job Token.
7. Map/Reduce Task report status to its Task Tracker, using Job Token.
8. Task Tracker sends heartbeats to Job Tracker, which includes status of all tasks. Job Tracker combines the output of Reducer Task.

Token summary

- Delegation token is created by Name Node used for Authentication with Name Node.
- Block Access Token created by Name Node used for Access Control with Data Node.
- Job Token is created by Job Tracker used for Authenticating with Task Tracker.

3.3 Authorization:

The task of which users can perform which actions with which resources is handled through authorization.

In plain Hadoop, HDFS file permission is based on traditional UNIX permission bits, each file has three permissions sets which include READ, WRITE, EXECUTE. Three different user classes will be defined including OWNER, GROUP, OTHERS. Based on the class user belongs to, HDFS sets access permission. For instance, if the user is the owner HDFS sets owner class permission. If the user is not the owner but is the member of the group class, HDFS sets the group class permission. If the user is not the owner or is not the member of the group class, then other class permission will be enforced. Although this model is sufficient for many organisations, it has some limitations. One of the limitations is that in the traditional UNIX model only one group is defined which the permission could be defined for that group. Thus, there will be a problem if there is a need to define more groups and set permission for each of them.

To solve this problem and provide more structured permission model, Portable Operating System Interface (POSIX) access control list is offered. Access Control list will provide the solution to define different permission models for different hierarchy of the users and groups. This will allow that for each file, different groups and users could have different permissions. In POSIX Access Control List, six types of Access Control List (ACL) entries will be defined. Each of these entries, defines permission for one user or a group of users.

- The "Owner" will define permission for the users who owns a file or directory.
- The "Owing Group" will define permission for the file's owning group.
- Every "Named User" entry defines permission for the user who is specified in the entry's qualifier list.
- Every "Named group" entry defines permission for the users who are specified in the entry's qualifier field.
- "Other" entry defines permission for all of the other users. [9]

Every Individual component with in Hadoop eco system implements their own Authorization module

- HDFS implements Authorisation at a file/folder level using UNIX POSIX model.
- HIVE (SQL on Hadoop) implements Authorisation on tables and columns just like RDBMS.

There are various tool sets that has its own security implementation within Hadoop eco system.

Which means if we want to do any Authorisation within Hive we have to go to hive tool and define security policies there, like for a particular table we want to define permission for this User Group. If we want to define in HDFS, we have to go to HDFS and define permission for a specific file/directory.

Like the same way Every individual Hadoop Tool set has its own security implementation, and also we need to go to individual components to define our security rules, when we talking about Hadoop as a data platform this is not a way to define security rules. We want all the security rules to be define at a single location.

This is exactly what **Apache Ranger** brings to Hadoop world.

3.3.1 Apache Ranger:

Apache Ranger provides a centralized platform to define, administer and manage security policies consistently across Hadoop Components [16]. In Apache Ranger instead of having individual components, Ranger Plugin is used. This Ranger Plugin will act as Authoriser within individual components.

How Ranger Works:

Ranger has

- Ranger Policy Store
- Ranger Audit Store

The Ranger Policy Store is a centralised place where we keep all our access control rules.

When a user is trying to access HDFS, instead of native HDFS ACL verification, it's going to use the Ranger Plugin to validate whether the user have access or not, and then writes the audit information to centralised Audit Store. This particular process will happen for individual Hadoop components. In Ranger there is one place user will go and define security policies and also all the audits come's back to one place, so instead of going to individual Audits of particular component to collect audits whether there is someone trying break in or not Ranger provides centralised Audit Store.

Ranger Components:

Ranger Admin Portal

Ranger Admin is the central interface for security Administration Ranger Admin Portal which is web based console which allows user to create and update policies. All of the policies defined actually get stored with in Ranger Policy Database

User Group Sync

Apache Ranger provides a user synchronization utility to pull users and groups from UNIX or from LDAP or Active Directory [17]. When we define any security policy, we are going to define for a specific set of user/groups, when we are defining for any user/group we don't want to make spelling mistakes or typo, we have to make sure that when giving permission for a particular user it has to be for right user, to do that, go and get all the users and groups that is present in corporate directory either it could be Light Weight Directory Access Protocol (LDAP) or Active Directory (AD) or from a UNIX Server and sync them back to Ranger Database that make sure selection of rigth users and right groups.

Ranger Plugin

Ranger plugin acts as an Authorizer within Name Node. Ranger plugins pull in policies from a central server and store them locally in a file. When a user request comes through the component, these plugins pull in policies from a central server and store them locally in a file. When a user request comes through the component, these plugins intercepts the request and evaluate it against the security policy. [17]

Ranger Policies for HDFS

Apache Ranger offers a federated Authorization model for HDFS. Ranger plugin for HDFS checks for Ranger policies and if a policy exists, access is granted to user. If a policy doesn't exist in Ranger, then Ranger would default to native permission model in HDFS (POSIX). [18]

3.4 Data Protection:

Encryption is a common method to protect data. There are two primary flavours of data encryption:

- Data-at-Rest Encryption
- Data-in-Transit Encryption

Data at rest refers to data that is stored even after machines are powered off. Data in transit refer to data on the move, such as data travelling on the internet, wifi, cell towers etc.

3.4.1 HDFS Data-at-Rest Encryption: [19]

HDFS implements transparent, end-to-end data encryption. .

Transparent Encryption

Data read, Data write to a special HDFS directory is transparently encrypted and decrypted without requiring changes to application code.

End-to-end Encryption

Data can only be encrypted and decrypted by the client.

Encryption Zone: [19]

For Transparent Encryption introduces a special directory called Encryption Zone, whose contents will be transparently encrypted upon write and transparently decrypted upon read. Encryption Zone is a directory in HDFS whose contents will be automatically encrypted on write and decrypted on read. Each encryption zone is associated with a single encryption zone key which is specified when zone is created. Each file within an encryption zone has its own unique data encryption key (DEK). The encryption zone key(EZK) is used to encrypt the DEK into an encrypted DEK (EDEK). The EDEK is then persisted as an extended attribute in the Name Node for a given file. EZK provides access to all data stored in encryption zone, to prevent access to EZKs and thus ability to decrypt any data, EZKs must not be stored in HDFS. EZKS need to be accessed through secure key server. The key server itself is a separate piece of software that handles the storage and retrieval of EZKs which needs to handle by a dedicated Hardware Security Module. In order

to have a separation of duties, there needs to be an intermediary between HDFS, HDFS client, key server. This is solved with introduction of Key Management Server (KMS).

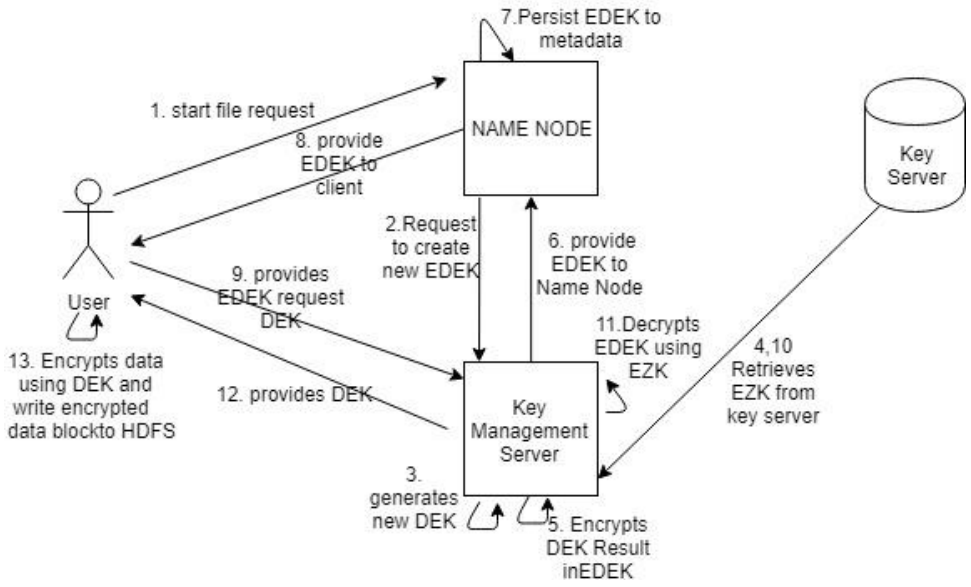
Encryption Zone key (EZK) + Data Encryption Key (DEK) = Encrypted Data Encryption Key (EDEK)

Hadoop Key Management Server (KMS):

Hadoop KMS is a cryptographic key management server. KMS acts as proxy on behalf of HDFS daemons and clients. In the context of HDFS Encryption KMS performs these responsibilities.

- Generating new encrypted data encryption keys for storage on the Name Node.
- Decrypting encrypted data encryption keys for use by HDFS clients.
- Providing access to stored encryption zone keys.

The KMS handles generating encryption keys EZKs, DEKs. Decrypting EDEKs, communicating with key server.

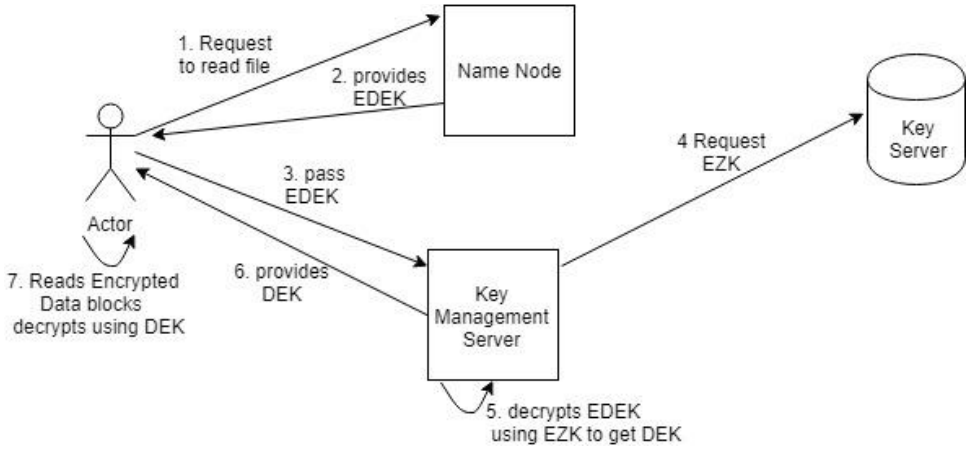


Writing files to HDFS Encryption Zone

HDFS client Writing to a new file stored in an Encryption Zone in HDFS: [20]

1. The HDFS client calls create() to write to the new file.
2. The Name Node requests the KMS to create a new EDEK.
3. The KMS generates a new DEK.
4. The KMS retrieves the EZK from the key server.
5. The KMS encrypts the DEK, resulting in the EDEK.
6. The KMS provides the EDEK to the Name Node.
7. The Name Node persists the EDEK as an extended attribute for the file metadata.
8. The Name Node provides the EDEK to the HDFS client.
9. The HDFS client provides the EDEK to the KMS, requesting the DEK.
10. The KMS requests the EZK from the key server.
11. The KMS decrypts the EDEK using the EZK.
12. The KMS provides the DEK to the HDFS client.
13. The HDFS client encrypts data using the DEK.
14. The HDFS client writes the encrypted data blocks to HDFS.

The sequence of events for Reading an encrypted file: [20]



Reading files from Encryption Zone

1. The HDFS client calls open() to read a file.
2. The Name Node provides EDEK to the client.
3. The HDFS client passes the EDEK to the KMS.
4. The KMS requests the EZK from the key server.
5. The KMS decrypts the EDEK using the EZK.
6. The KMS provides the DEK to the HDFS client.
7. The HDFS client reads the encrypted data blocks, decrypting them with the DEK.

All of the above steps for the read and write path happen automatically through interaction between the client, Name Node, KMS. Access to encrypted file data and meta data is controlled by normal HDFS file system permission. This means if any malicious user access HDFS he can gain access to cipher text and encrypted keys. How ever since access to encryption zone keys is controlled by a separate set of permissions on the KMS and key store, this does not pose a security threat.

3.4.2 Hadoop Data-in-Transit Encryption: [20]

Hadoop has several methods of communication over the network

Hadoop RPC Encryption

These are performed by clients using the Hadoop API, by Map Reduce, among the Hadoop services Name Node, Data Node, Job Tracker, Task tracker [21], Hadoop RPC system implements Simple Authentication and Security Layer (SASL).

Hadoop data transfer protocol Encryption [20]

When HDFS data is transferred from one Data Node to another or between Data Nodes and their clients, a direct TCP/IP socket is used in a protocol known as the HDFS data transfer protocol. The Hadoop RPC protocol is used to exchange an encryption key for use in the data transfer protocol when data transfer encryption is enabled.

Hadoop HTTP Encryption [20]

There is a well-known and proven method to encrypt the data in transit using HTTPS, which is an enhancement of HTTP with SSL/TLS. Hadoop uses HTTP for its Web UIs, for the Map Reduce shuffle phase.

3.5 Protecting the Hadoop Cluster:

The Logical Network Segmentation operates at the network layer using Internet Protocol (IP) addressing. With logical separation, devices in same network segment are grouped together. The logical separation is achieved through the use of network subnets. If a Hadoop cluster has 150 nodes, these nodes are logically grouped on the same subnet. Organizing hosts logically makes it easy to administer and secure. Most common method of implementing network segmentation is through the use of Virtual Local Area Networks (VLANs).

After Segmenting the Hadoop Cluster, the segment has to be protected. This is achieved with 'network firewalls', 'intrusion detection and prevention'.[20]

Network Firewall

A network firewall protects an entire network from incoming intrusions. Network firewalls guards an internal computer network against malicious access. When a fire wall is used it constantly monitors all incoming and outgoing traffic and allows or drop network packets based on network properties.

Intrusion Detection and prevention

The network firewalls control flows into and out of the network from a Hadoop cluster. What happens if a malicious attacker has bypassed the network firewall and attempting exploits against machines in the cluster, such as buffer overflow attacks, distributed denial-of-service attacks. Intrusion detection and prevention systems can help stop these types of attacks.

The Knox gate way simplifies Hadoop security for users that access the cluster data and manages the cluster. The gate way run's as a server provides centralized access to Hadoop cluster. Knox hides the internal Hadoop cluster Topology from potential attackers. Limits the network endpoints (firewall holes)

What is Apache Knox what it provides?

Secure entry point for Hadoop clusters. The Knox or Knox gateway is a proxy for interacting with Apache Hadoop cluster in a secure way providing Authentication to secure any HTTP interactions in the cluster. Knox supports LDAP, Active Directory, Single Sign on Authentication systems.

How Apache Knox differ from Kerberos?

Apache Knox is not an alternative to Kerberos, Knox is a security perimeter behind firewall and provides single point of access to underlying Hadoop services. Apache Knox integrates with identity management and a user will authenticate to Knox. Once user is authenticated with Knox then uses Kerberos to Authenticate with other Hadoop services securely.

Knox gateway controls all Hadoop REST API access through firewall. The goal is to simplify Hadoop security for the users who access the cluster data and execute jobs, for the users who control access and who manage the cluster. [22]

Apache KNOX services:

- **Proxying Services:** provide access to Apache Hadoop via proxying or HTTP Resources
- **Authentication Services:** Authentication for REST API, Authentication via Light Weight Directory Access Protocol/Active Directory, Kerberos, Security Assertion Mark-up Language Authentication etc.
- **Client Services:** Client Development can be done with scripting or using KNOX shell.

Coupled with Kerberos secured Hadoop Cluster, KNOX Gateway provides enterprise with a solution that,

- Integrates with enterprise Identity Management
- Simplifies the number of services that client needs to interact with
- Protects the details of cluster deployment. [28]

3.6 Auditing:

Auditing and monitoring are critical to data security. Through Audit we can ensure that the security controls that are in place are working correctly and identify attempts to find a way around. To record the actions logs are the common methods which allows administrators and auditors to review a user's actions. Logs provides evidence of transaction performed. [23]

Hadoop has many different components, for each component there is audit log.

HDFS Audit Logs

HDFS has two different audit logs

- Hdfs-audit.log for user activity.
- SecurityAuth-hdfs.audit for service activity.

Both of these logs are implemented with Apache Log4j.

Map Reduce Audit Logs

Like HDFS Map Reduce has two logs

- Mapred-audit.log for user activity
- SecurityAuth-mapred-audit.log for service activity

For every Hadoop component there is an Audit log, when we talking about Hadoop as a data platform this is not a way to define Audit logs. We want all the Audits to be define at a single location.

Apache Ranger provides centralised Audit Mechanism, all essential for Hadoop. Ranger Audit framework supports saving audit logs to RDBMS for faster analysis and in HDFS for permanent storage. Audit log which are stored can later be processed by other applications to query and reporting.

When user enables Audit to HDFS, it helps user to view logs, the kind of information collected in logs is based on the user activity, for example if any unauthorized user trying to access HDFS file without Authorization permissions, the process fails and the user activity is logged into HDFS.

Audit to HDFS Schema:

Id, result, access, enforcer, clip, policy, repo, repoType, reason, evtTime, reqUser, action, resource, resType, seq_num, event_count, event_dur_ms, tags, agentHost, logType.

For every action performed by the user there is an entry into Audit log, which shows information based on the schema.

Example: User try to create directory which has only READ Permission (I.e access = READ)

hdfs dfs –mkdir /user/abc

The result of the command is, mkdir: permission denied: access=WRITE

If we see the audit log for the above action, it shows the values (according to schema)

Id: 123456, Result: 0 (denied), access:READ, date, time, ….....

Audit log records the following information

- Successful or UnSuccessful, user Authentication at Knox Gateway.
- User activity for reading/writing a HDFS file
- User activity for map/reduce execution etc.

Audit logs provides an evidence if any user engages in unauthorized activity. It provides more efficient way to observe the user actions and keep the data more secure.

Monitoring and Analysis:

After setting up all the Hadoop logging, the important step is monitor the cluster for security events, breaches, suspicious activity. By using logging information Hadoop can use for security analytics, Advance persistence threat analytics and user behaviour machine learning built on Hadoop. This Audit log information can be used for evaluation in the form of Audit Analysis Report

4.Security Flow to Protect Data in the Cluster

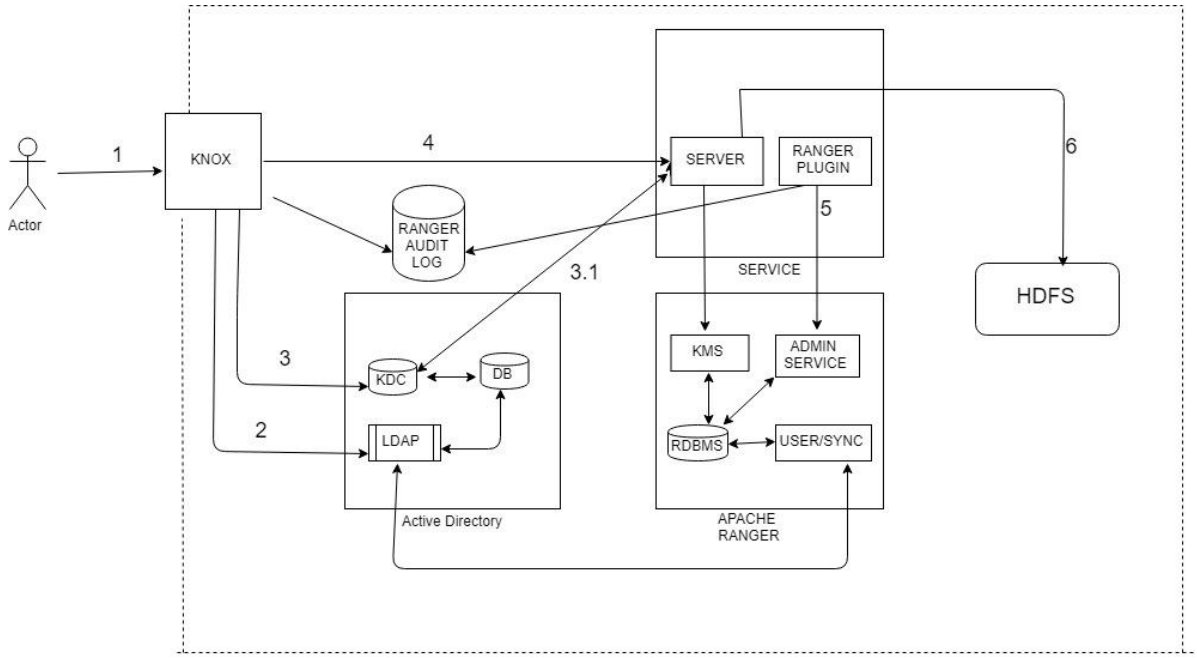
4.1 Security Flow:

Security flow based on five key principles Authentication, Authorization, Data protection, Protecting the Network, Auditing.

1. Initially Hadoop cluster is insecure, cluster services need to authenticate to make it secure, Implement Kerberos to perform Authentication services between Hadoop daemons, implementing Kerberos is like issuing passports to Hadoop users which establish their true identity. Kerberos mode in Hadoop ecosystem provides a secure Hadoop environment.
2. Deploy Apache Ranger, ranger centralized interface defines and enforces access policies it was like issuing different types of visas allowing the users to access various parts of ecosystem. Ranger is used for deciding who can access what resources on a Hadoop cluster with the help of policies. Ranger acts as Authorization system which allows / denies access to Hadoop cluster resources like HDFS files, Hive Tables etc, based on pre-defined Ranger policies. When user request comes to Ranger, it is assumed to be Authenticated already.
3. Encrypt the data, make it unreadable while flowing through the network (data-in-motion) or stored on disk (data-at-rest) in Hadoop. For data-at-rest HDFS implements Transparent Data Encryption where data reads and writes to a special HDFS directory called encryption zone. When encryption zone is enabled the data is automatically encrypted on write and automatically decrypted on read. Hadoop http encryption is a proven method to encrypt the data-in-motion using HTTPS.
4. To Protect the Hadoop Cluster, implement Apache Knox, a centralized gateway to Hadoop cluster. Knox Validates users passports and visas before letting them in the system Knox can be imagined as the gatekeeper which decides whether to allow user access to Hadoop cluster or not. Knox is a Rest API based perimeter security gateway system which 'Authenticates' user credentials (mostly against AD/LDAP) only successfully Authenticated user are allowed access to Hadoop cluster.
5. Audit the events and monitor for security analysis, use Apache Ranger for Auditing which provides centralised Audit mechanism and monitor the cluster for security events. When Auditing is enabled it helps the user to view logs, it provides an evidence for any unauthorization activity, by seeing the logs user can monitor the cluster for security events, breaches etc. The audit log information can be used for evaluation in the form of Audit Analysis Report.

4.2 Example Security flow using Apache Knox

When a user wants to push a file into HDFS



Security Flow

1. User sends a HDFS request including the file and the command to put that file into the desired directory to KNOX.
2. Before entering into the cluster KNOX Authenticates the user by validating username/password via LDAP provider, this event is logged into Audit Log.
3. If the Authentication is successful at KNOX Gateway, KNOX acts as proxy user and initially gets service ticket from Key Distribution Center for HDFS file storage and Authenticates at Name Node.
4. Knox request the service to perform the action i.e to store the file into HDFS as a user.
5. Ranger HDFS plugin checks for Authorization, if user has the permission to write to desired directory. If the user doesn't have enough permission the process ends here. This event is logged into Ranger Audit Log.
6. If the user has enough permissions to store the files, file is pushed to HDFS. If data-at-rest encryption is enabled data is pushed to HDFS encryption zone. [24]

Conclusion

Due to increase of new technology, social networking sites, usage of devices, the amount of data produced is growing rapidly with huge volume, high velocity and variety of data which includes structured, unstructured and semi structured data. One popular platform used to store large amount of data is Hadoop. When Hadoop was initially designed security was not much considered. Security Threats were identified in Hadoop like unauthorized user can access HDFS files, eavesdrops data packets sent by Data Nodes to client etc. There is no protection for data-at-rest, data-in-motion. Hadoop is not properly addressed by firewalls. As security plays an important role in keeping the sensitive data protected the goal is to evaluate the security of Hadoop, data needs to be protected in order to avoid security breaches. Different projects have started to improve the security of the Hadoop, in this thesis the security of the traditional Hadoop is evaluated based on five key principles of Information security. Authentication make sure the user who he claims to be, this project uses Kerberos to Authenticate client and Hadoop services, it make sure only a proper Authenticated user can communicate successfully with a kerberized Hadoop service. Authorization manage access to resources, this project implements Apache Ranger which provides a centralized platform to define security policies consistently. Data protection protect against leakage of data, this project implements Transparent Data Encryption for HDFS data-at-rest, the contents will be automatically encrypted on write and decrypted on read. Protecting the Hadoop cluster, this project implements Apache Knox for interacting with Hadoop cluster in a secure way, Apache Knox acts as a secure entry point for Hadoop cluster. Auditing allows monitoring the user activity and analysis to prevent security breaches, this project implements Apache Ranger Audit Framework which provides centralized security for Hadoop ecosystem. This Thesis suggests a security flow based on Information Security Principles in order to protect the data in the cluster.

Future Work:

Evaluate the security system in Hadoop version2 in more details, based on the Information security principles. Key area to consider could be to work on securing the yarn services.

BIBLIOGRAPHY

[1] T.White, Hadoop: The Definitive Guide: O'Reilly Media, 2009

[2] Sudipta Chandra, Soumya Ray, R.T.Goswami: Big Data Security Survey on Frameworks and Algorithms: 2017 IEEE 7th International Advance Computing Conference

[3] JOHN R. VACCA: Cloud Computing Security FOUNDATIONS AND CHALLENGES: CRC Press,2016

[4] Cloud Security Alliance (CSA): The Treacherous 12: Top Threats Working Group: Cloud Computing Top Threats in 2016-17: https://downloads.cloudsecurityalliance.org/assets/research/top-threats/Treacherous-12_Cloud-Computing_Top-Threats.pdf

[5] Tutorials point: Hadoop Distributed File Systems: https://www.tutorialspoint.com/hadoop/hadoop_introduction.htm

[6] Name Node Hadoop Wiki: https://wiki.apache.org/hadoop/NameNode

[7] Data Node Hadoop Wiki: https://wiki.apache.org/hadoop/DataNode

[8] Hadoop advantages and disadvantages: http://blogs.mindsmapped.com/bigdatahadoop/hadoop-advantages-and-disadvantages/

[9] Evaluation of Security in Hadoop: MAHSA TABATABAEI: Master's Degree Project Stockholm, Sweden, 2014.

[10] security by design principles: https://www.owasp.org/index.php/Security_by_Design_Principles

[11] Analytics and information management: https://www2.deloitte.com/content/dam/Deloitte/de/Documents/technology/TECH-Hadoop-Best-Practices-Security-Big-data-platform-large-enterprises-2017.pdf

[12] Hadoop and Kerberos https://steveloughran.gitbooks.io/kerberos_and_hadoop/content/sections/hadoop_and_kerberos.html

[13] Kerberos work flow image: https://www.google.it/search?q=kerberos+work+flow&source=lnms&tbm=isch&sa=X&ved=0ahUKEwj57fKfus7ZAhVDOBQKHaHbAk0Q_AUICigB&biw=1366&bih=637#imgrc=KaPpApbSXJkL6M:

[14] ravi's tech blog: https://ravistechblog.wordpress.com/tag/hadoop-an-kerberos/

[15] moving to the cloud https://books.google.it/books?id=HaHz3-fqGM0C&pg=PA296&lpg=PA296&dq=namenode+as+kdc&source=bl&ots=4iwvXxDuq5&sig=OMLI8neVNvsZeibvtpA-63eOwwE&hl=en&sa=X&ved=0ahUKEwjpqoLP7cXZAhWBMRQKHXdYAlAQ6AEIejAI#v=onepage&q=namenode%20as%20kdc&f=false

[16] Ranger overview: https://hortonworks.com/apache/ranger/

[17] Ranger Components: https://hortonworks.com/apache/ranger/#section_2

[18] Ranger best practise: https://hortonworks.com/blog/best-practices-in-hdfs-authorization-with-apache-ranger/

[19] Transparent encryption in hdfs: https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/TransparentEncryption.html

[20] Ben Spivey & Joey Echeverria: Hadoop Security: Protecting your Big Data Platform: O'Reilly Media, 2015

[21] cloudera: http://blog.cloudera.com/blog/2013/03/how-to-set-up-a-hadoop-cluster-with-network-encryption/

[22] Knox Hadoop: https://hortonworks.com/blog/introducing-knox-hadoop-security/

[23] logs: https://www.securityweek.com/hadoop-audit-and-logging-back-time

[24] Knox example: https://community.hortonworks.com/articles/102957/hadoop-security-concepts.html

[25] cloudera blog: http://blog.cloudera.com/blog/2012/03/authorization-and-authentication-in-hadoop/

[26] Kerberos wiki: https://en.wikipedia.org/wiki/Kerberos_(protocol)

[27] ibm: https://www.ibm.com/support/knowledgecenter/en/STXKQY_4.2.0/com.ibm.spectrum.scale.v4r2.adv.doc/bl1adv_thekerberosmode.htm

[28] https://knox.apache.org/

[29] https://blog.cloudera.com/blog/2017/12/hadoop-delegation-tokens-explained/