Scientific Journal of Impact Factor (SJIF): 4.14

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 3, Issue 5, May -2016

Network Level Security In Hadoop Using WireLine Encryption

Prof S.A.Darade[#1] Kiran Kamble[#2] Gauravi Khanapure[#3] Sneha Chavan[#4] Komal Kumbhar[#5]

[1,2,3,4,5] Computer Engineering, Sinhgad Institute of Technology & Science

Abstract -*Hadoop is a open-source framework and a distributed system that provides a Distributed file system(HDFS) and MapReduce. Hadoop is used on private clusters and is use to store sensitive data; as a result strong authentication and authorization is necessary to protect private data. Adding security to Hadoop is challenging because all the interactions do not follow the classic client-server architecture. To address these security challenges, the Keberoes authentication mechanism is supplemented by delegation and capability like access tokens. A huge number of companies have begun using the open source technology as Apache Hadoop Distributed File System to store and analyze large volume of structured and unstructerd data which are captured from social media networks, websites, etc. There are several ways a user access the data on Hadoop clusters. The goal is to explore some advanced methodologies which deals with security and privacy concern for people who out-source data on Hadoop clusters.*

Keywords *- Big Data, Hadoop, HDFS, MapReduce, NameNode, DataNode.*

I. INTRODUCTION

Big Data is current and widely used technology, it holds large tera bytes of data which cannot be maintained and stored in traditional database. There are many issues related to Big Data viz, management issues, processing issues, security, security issues and storage issues. Hadoop (Highly Archived Distributed Object Oriented Programming) is an Open source Java Framework technology helps to store, access and gain large resources from big data in a distributed form gaining less cost,high degree of fault tolerance and high scalability. The Hadoop has two types of nodes operating in master and slave pattern: a NameNode and DataNode. The NameNode manages and maintains the file system and its metadata. It is the master of the HDFS which tracks the blocks in DataNode and if any block failed in replica of DataNode the NameNode creates another replica of the same block and also reduces the data loss. The DataNode performs the work of HDFS like read, write, move, copy on the local system. Hadoop contains sensitive data and it is prone to security issues due to no appropriate role based access for it.

The replicated data is not secured which needs more protection from data breaches and vulnerabilities. Mostly Government sectors never use Hadoop for storing valuable and sensitive data as it need to provide external security such as firewalls and Intrusion Detection System. Hadoop environment provides security by using encrypting the block levels and individual file system by using encryption technique but yet no algorithm is designed for maintain security in Hadoop.

II. LITERATURE SURVEY

Hadoop environments can include various variety and sensitive data. Collection of data into one environment also increases the risk of data stealth and accidental revelation of data. The technology to collect and store data from multiple sources can create a elevation of problems related to currently used access control and management. It is felt that, Kerberos and current Access control listsused alone are not adequate for enterprise needs [2]. Thus it has been noticed that due to data Security, access control and lack of role-based access are part of the reason why Hadoop is not an alternate option for relational database in the enterprise.

Kerberos is the network authentication protocol which allows the node to transfer any file over non secure channel by a ticket to prove their unique identification between them. This Kerberos mechanism is used to reinforce the security in HDFS. In HDFS the connection between Name node and client is achieved using Remote Procedure Call and the connection from Client (client uses HTTP) to Data node is Achieved using Block Transfer. Here the Kerberos or Token is used to authenticate a RPC connection. If the Client needs to obtain a token means, the client makes use of Kerberos Authenticating Connection. Ticket Granting Ticket (TGT) or Service Ticket are used to authenticate a name node by using Kerberos. Both ST and TGT can be renewed after long running of jobs while Kerberos is renewed, new ST and TGT is also issued and distributed to all task. The Key Distribution Centre (KDC) issues the Kerberos Service Ticket using TGT on receiving the request from task and network traffic is avoided to the KDC by using Tokens. Thus keeping the ticket constant the time period is extended in Name Node.

The major advantage is if the ticket is abducted by the attacker it can't be regenerated. We can also use another method for providing security for file access in HDFS [1].

III. PROPOSED METHODOLOGY

We have proposed new method to secure data at HDFS by analyzing all older methods. It is implemented by using Wire Line Security (called Open Standard for Authorization) using Real Time Encryption Algorithm.We are using an Open Authentication Protocol that helps to run-over the problems of conventional clientserver authentication model. In the conventional client-server model, the client requests to an access protected resource on the server by authenticating itself using the resource owner's passport. In order to give third-party applications access to restricted resources, the resource owner verifies its authorization with the third-party. In proposed system wire line security is used to authenticate user and it also return unique token for each user who attempt successful login. The token returned by Auth server used in encryption method so it provides data confidentiality and integrity to the user data. The files are encrypted before load to HDFS and decrypted when job execution is in progress . The Real Time Encryption Algorithm use the Auth token as key and Encrypt data by XoRing with the key.

Algorithm:

1. First, the user authenticates with the EIM system using the user credentials.
2. The EIM system issues the Kerberos ticket to the user after authentication.
3. Then the user presents this ticket to Hadoop to perform operations on the secured Hadoop cluster. The Hadoop daemons trust the EIM system, issue ticket due to the cross-realm trust established between Hadoop local KDC and the EIM system.
4. The Hadoop daemon fetches the user group information from LDAP to provide the authorized access to the user. If the user IDs and the Kerberos principals are not the same, the mapping of the user ID to Kerberos principal is defined in the core-site.xml.
5. To ensure that there is a centralized management of user credentials and roles, there is a need to synchronize the user groups between the EIM system and the local KDC. Only the roles and groups are synchronized and user credentials are stored only in the EIM system.
6. To ensure that the Hadoop daemons authenticate the end user using the Kerberos ticket issued by the EIM system, we need to establish the cross-realm trust between the Hadoop local KDC and the EIM system.

Note: Enterprise Identity Management (EIM)

IV. EXPERIMENTAL SETUP

To carry out the experiment we have installed Ubuntu Linux 12.04. Openjdk1.7 installed in it and SSH enabled. Hadoop 1.2.1 have been configured as a Single-Node Cluster to use the HDFS and Map Reduce capabilities. The Name Node is centre piece of Hadoop in light of the fact that it controls the entire Data Nodes exhibit in bunch. It is a Single-Point-of-Failure yet late forms accompany Backup Name Node to make it exceptionally accessible. The Data Nodes contain all the information in bunch on which we will work our Map Reduce projects and perspective the activity information from different points of view. Job Tracker controls all the tasks which are running on Task Trackers shown in following figure:

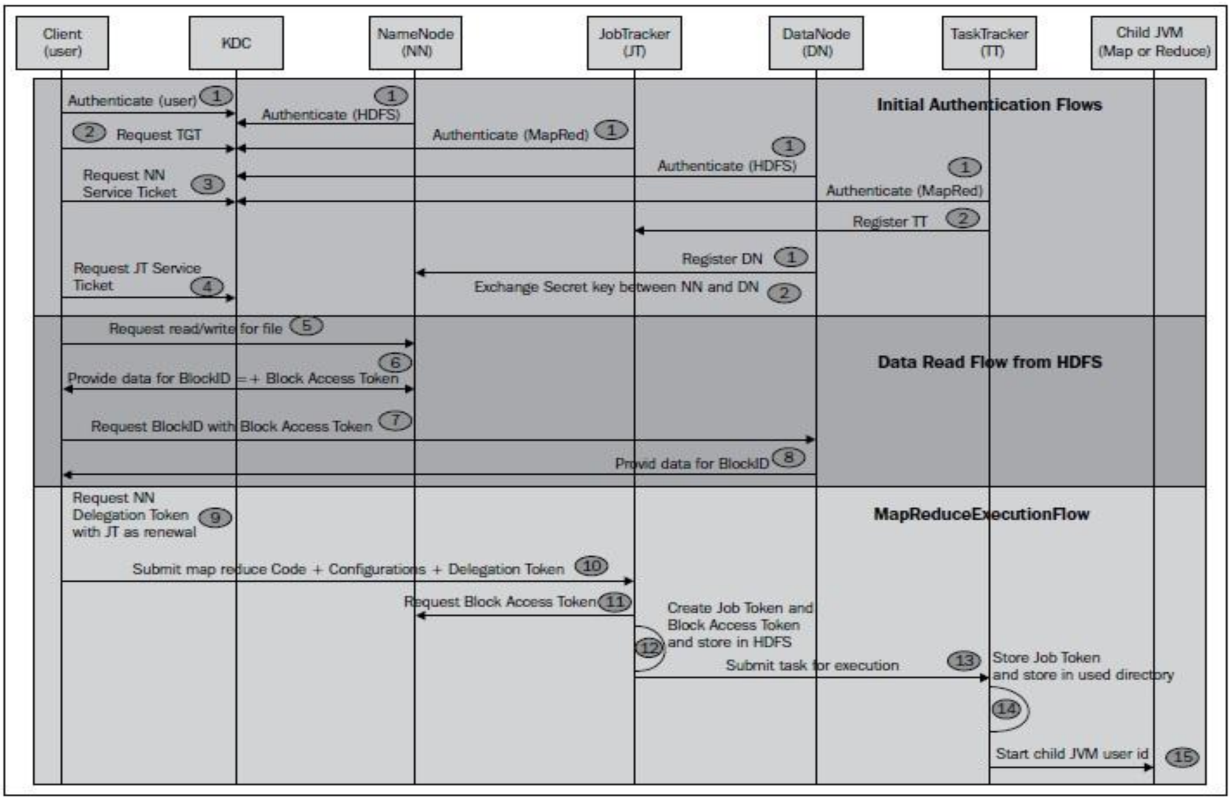
Client (user)
KDC
NameNode (NN)
JobTracker (JT)
DataNode (DN)
TaskTracker (TT)
Child JVM (Map or Reduce)
Initial Authentication Flows
Authenticate (user) 1
Authenticate (HDFS) 1
2 Request TGT
Authenticate (MapRed) 1
Authenticate (HDFS) 1
Request NN Service Ticket 3
Authenticate (MapRed) 1
Register TT 2
Register DN 1
Request JT Service Ticket 4
Exchange Secret key between NN and DN 2
Request read/write for file 5
Data Read Flow from HDFS
Provide data for BlockID + Block Access Token 6
Request BlockID with Block Access Token 7
Provid data for BlockID 8
Request NN Delegation Token with JT as renewal 9
MapReduceExecutionFlow
Submit map reduce Code + Configurations + Delegation Token 10
Request Block Access Token 11
Create Job Token and Block Access Token and store in HDFS 12
Submit task for execution 13
Store Job Token and store in used directory 14
Start child JVM user id 15

Fig1.Sequence Diagram

V. IMPLEMENTATION

To Create the Kerberos keytab files

The instructions in this section for creating keytab files require using the Kerberos norandkey option in the xst command. If your version of Kerberos does not support the norandkey option, or if you cannot use kadmin.local, then use these alternate instructions in Appendix F to create appropriate Kerberos keytab files. After using those alternate instructions to create the keytab files, continue with the next section To deploy the Kerberos keytab files. Do the following steps for every host in your cluster. Run the commands in the kadmin.local or kadmin shell, replacing thefully.qualified.domain.name in the commands with the fully qualified domain name of each host:

1. Create the hdfs keytab file that will contain the hdfs principal and HTTPprincipal. This keytab file is used for the NameNode, Secondary NameNode, and DataNodes.
   kadmin: xst -norandkey -k hdfs.keytab hdfs/fully.qualified.domain.name HTTP/fully.qualified.domain.name
2. Create the mapred keytab file that will contain the mapred principal and HTTPprincipal. If you are using MRv1, the mapred keytab file is used for the JobTracker and TaskTrackers. If you are using YARN, the mapred keytab file is used for the MapReduce Job History Server.
   kadmin: xst-norandkey-kmapred.keytab
   mapred/fully.qualified.domain.name HTTP/fully.qualified.domain.name
3. YARN only: Create the yarn keytab file that will contain the yarn principal andHTTP principal. This keytab file is used for the ResourceManager and NodeManager.
   kadmin: xst -norandkey -k yarn.keytab yarn/fully.qualified.domain.name HTTP/fully.qualified.domain.name
4. Use klist to display the keytab file entries; a correctly-created hdfs keytab file should look something like this:
5. $ klist -e -k -t hdfs.keytab
6. Keytab name: WRFILE:hdfs.keytab
7. slot KVNO Principal
8. ---- ---- ---
9. HTTP/fully.qualified.domain.name@YOUR-REALM.COM(DES cbc mode with CRC-32)
10. HTTP/fully.qualified.domain.name@YOUR-REALM.COM(Triple DES cbc mode with HMAC/sha1)
11. hdfs/fully.qualified.domain.name@YOUR-REALM.COM(DES cbc mode with CRC-32)

hdfs/fully.qualified.domain.name@YOUR-REALM.COM(Triple DES cbc mode with HMAC/sha1)

12. Continue with the next section To deploy the Kerberos keytab files.To deploy the Kerberos keytab files

On every node in the cluster, repeat the following steps to deploy the hdfs.keytab andmapred.keytab files. If you are using YARN, you will also deploy the yarn.keytabfile.

1. On the host machine, copy or move the keytab files to a directory that Hadoop can access, such as /etc/hadoop/conf.

a. If you are using MRv1:
$ sudo mv hdfs.keytab mapred.keytab /etc/hadoop/conf/
If you are using YARN:
$ sudo mv hdfs.keytab mapred.keytab yarn.keytab /etc/hadoop/conf/

b. Make sure that the hdfs.keytab file is only readable by the hdfs user, and that the mapred.keytab file is only readable by the mapred user.

c. $ sudo chown hdfs:hadoop /etc/hadoop/conf/hdfs.keytab

d. $ sudo chown mapred:hadoop /etc/hadoop/conf/mapred.keytab
$ sudo chmod 400 /etc/hadoop/conf/*.keytab

Note: To enable you to use the same configuration files on every host, Cloudera recommends that you use the same name for the keytab files on every host.

e. YARN only: Make sure that the yarn.keytab file is only readable by theyarn user.

f. $ sudo chown yarn:hadoop /etc/hadoop/conf/yarn.keytab
$ sudo chmod 400 /etc/hadoop/conf/yarn.keytab Important:

If the NameNode, Secondary NameNode, DataNode, JobTracker, TaskTrackers, HttpFS, or Oozie services are configured to use Kerberos HTTP SPNEGO authentication, and two or more of these services are running on the same host, then all of the running services must use the same HTTP principal and keytab file used for their HTTP endpoints.

VI. CONCLUSION

The security issue is pointed more in order to increase the security in big data. We can improve security in big data by using the Wire Encryption approach in Hadoop Distributed File System which is the base layer in Hadoop, where it contains large number of blocks. This approach is introduced to overcome certain issues occurs in the name node and also in Data node. In Future, these methodologies can also implement in other layers of Hadoop Technology.

VII. REFERENCES

[1] B. Saraladevia, N. Pazhanirajaa, P. Victer Paula, M.S. Saleem Bashab, P. Dhavachelvanc "Big Data and Hadoop- A Study in Security Perspective", 2nd International Symposium on BigData and Cloud Computing(ISBCC'15)2015.

[2] Rajesh Laxman Gaikwad, Prof. Dhananjay M Dakhane and Prof. Ravindra L Pardhi "Network Security Enhancement in Hadoop Clusters", Volume 2, Issue 3, March 2013.

[3] Prashat D. Londhe, Satish S. Kumbhar, Ramakant S. Sul, Amit J. Khadse "Processing Big Data Using Hadoop Framework" 2014.

[4] Venkata Narasimha Inukollu , Sailaja Arsi and Srinivasa Rao Ravuri "High Level View Of Cloud Security: Issues And Solutions"

[5] Priya P. Sharma, Chandrakant P. Navdeti "Securing Big Data Hadoop: A Review of Security Issues, Threats and Solution" , Volume 5(2) 2014.