Secure HDFS Using OAuth 2.0

Ms. Anuja S. Paval, Prof. Amol S. Dange

M. E. Student, CSE Dept., Annasaheb Dange College of Engg.&Tech.Ashta; India;
Professor, CSE Dept., Annasaheb Dange College of Engg.&Tech. Ashta; India;

Abstract *—To Analyse big data is a very challenging task. Hadoop framework which is properly used for processing large amount of data on its distributed programming framework. When Hadoop was invented it was without any security model. So for security purpose we are implementing encryption scheme. For better result we designed a Real Time Encryption Algorithm(RTEA). We compared with Advanced Encryption Algorithm(AES) which is standard algorithm. Basic purpose of study is security. For security purpose we can use encryption and decryption before writing and reading data. For more security Kerberos can be used for authentication various ways of securing the data in hadoop framework are authentication, authorization and encryption of data.*

Keywords — *AES, DataNode, Hadoop, HDFS, OAuth.*

I. INTRODUCTION

Hadoop is open source java based programming framework. Hadoop was developed from GFS (Google File System)[8]. It is very popular, because of its highly scalable distributed programming framework; it enables processing big data for data-intensive applications. Hadoop is a framework of tools. Hadoop supports running application on big data. It provide MapReduce programming architecture with a Hadoop distributed file system(HDFS).It has massive data processing capability with thousands of commodity hardware by using simply its map and reduce functions. With the growth of the world, the data also increased and the storage space requirement also increased. Due to this there various challenges and issues comes out that should be handled. The world with growth generates the massive and complex data. That defines the basic concept of the Big Data. The computer science goal is to extract the information from the massive and complex data sets by the some analysis and technologies and use that information for decisions [7].

A. Security Risks in HDFS

The following are the threats which can be introduced in Hadoop.

1) Using RPC or HTTP protocols, an unauthorized user may access HDFS file and could execute arbitrary code.

2) An unauthorized user can read/write a data block of a file at a DataNode via the pipeline streaming Data-transfer protocol.

3) An unauthorized user may gain access privileges and may submit a job to a queue or delete or change priority of the job.

4) An unauthorized user may access intermediate data of Map job via its task trackers HTTP Shuffle protocol.

5)DataNodes does not have access control, unauthorized user could read arbitrary data blocks from DataNodes, or can write garbage data to DataNode.

Hadoop Distributed File System(HDFS) :

In Hadoop distributed file system, actual data is stored on DataNodes&NameNode contains the metadata & edit log. Files splits into blocks of equal size except last block & these blocks are then replicated across DataNodes. Where block size & replication factor are configurable parameters. MapReduce is a program model for distributed computing & it contains two important tasks which are Map and Reduce.

Secure Hadoop :

Hadoop usually deals with large volumes of data & encryption/decryption takes time, it is important that the framework used should perform encryption/decryption fast enough, So that it does not impact performance.

To build an infrastructure that is cost effective & efficiently scalable to meet customers requirements, It is a need to share storage devices & physical devices between multiple users, this is called Multitenancy.

One of the Solution to overcome security issue is to encrypt the data.

II. PRPOSED SYSTEM

OAuth 2.0 is an Open Authentication Protocol that helps to run-over the problems of conventional client-server authentication model. In the conventional client-server model, the client requests to an access protected resource to the server for authenticating itself. To give the applications access to the third-party for restricted resources, the resource owner verifies its authorization with the third-party.

User of Hadoop registered with system using the OAuth server, it generates two different types of tokens which used for different purposes.

When user logins, it is authenticated by authentication token and authorization token used in encryption and decryption to maintain data privacy amongst different users.

System Architecture shown in figure 1, User login in to system via web server but it verify user identity by using OAuth Server; it verify user and generate token id then user input file to load at HDFS through web server but before load to HDFS, it will send that data to data encryption model which will process the data and load at HDFS. When user submit job execution request then data decryption model decrypt requested data first and then sent it to MapReduce programming model. OAuth provide authentication token which is used for user verification and authorization token to build Random Key Generator Table which is useful to generate random and unique key for each user which is used in encryption and decryption process.

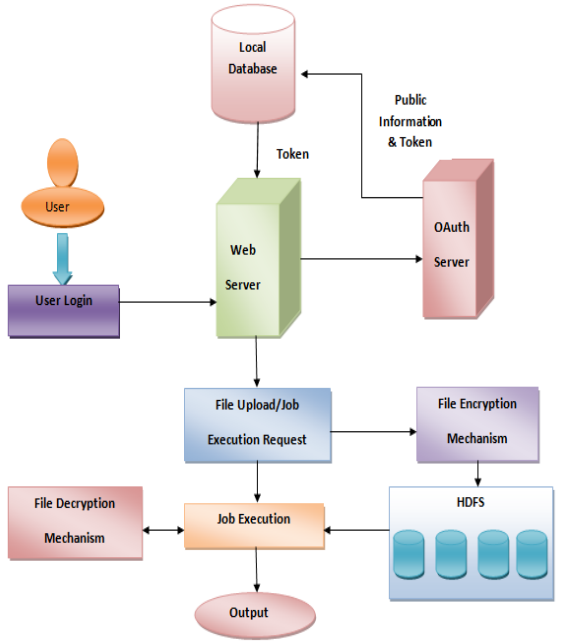

Figure 1 System Architecture

Real Time Encryption Algorithm

Encryption Steps :

1. Start
2. Retrieve OAuth token at successful user login
3. Generate random key using key generator
4. Read data from file and XoRing with the key
5. Add key in the XoRed data, which generated by key generator
6. Write encrypted data in file and load file to HDFS
7. Stop

Decryption Steps :

1. Start
2. Retrieve data to decrypt
3. Extract key from data using key generator
4. Read data from file and XoRing with the key
5. Pass decrypted data to MapReduce
6. Stop

Mathematical Model using Set Theory

1. Let S= {} be as a secure Hadoop system
2. Obtain an OAuth authentication tokens AT
   AT = {uid_at1, uid_at2....., uid_atn}
   Where uid_at1= unique token for specific user.
   S= {AT}
3. Obtain an OAuth authorization tokens OT
   OT = {uid_ot1, uid_ot2,.....,uid_otn}
   Where uid ot1= unique token for specific user.
   S= {AT, OT}
4. Give input files upload to HDFS F
   F= {f1, f2 ...fn}
   Where f1is a text file S= {AT, OT, F}
5. Perform encryption process on set of files is a En
   En= {F, OT}
   Where En process take input as set of files & user authorization tokenS = {AT, OT, F, En}
6. Perform decryption process on set of files is a Dn
   Dn = {F, OT}
   Where Dn process take input as set of files & user authorization token S = {AT, OT, F, En, Dn }
7. Identify MapReduce job to analyze data at HDFS
   J = {j1_dn,j2_dn,.....,jn_dn}
   Where j1_dn is a MapReduce program with decryption process S = {AT, OT, F, En, Dn, J}
8. Final Set S = {AT, OT, F, En, Dn, J}

Mathematical Model for proposed system

1. Initialize Tokens
   A) At = {}
   B) Ot = {}
2. Initialize path/files upload to HDFS F = {}
3. Process encryption module En= fp, uid_otn
   Where $fp \in F$
   $uid\_otn \in OT$
4. Execute job J = Fc, uid_otn Where $Fc \in En$
5. Encrypted files obtained by equation

$$S(En) = \sum_{n+1}^{fn} \text{fp\^uid\_ot}$$

Where n is total number of files in a file set F={}, fp is the plain text file and uid_ot is a user Authorization token

6. Job execution obtained by equation

$$S(Dn) = \sum_{n+1}^{fn} \text{fc\^uid\_ot}$$

Where n is total number of files in a file set F={} fc is the cipher text file and uid_ot is a user Authorization token

III. EXPERIMENTAL SETUP

To carry out the experiment we have installed Ubuntu Linux 16.04. Openjdk1.7 and Apache Tomcat 1.7 installed in it and SSH enabled. Hadoop 1.2.1 have been configured as a Single-Node Cluster to use the HDFS and MapReduce capabilities. To setup OAuth server we deploy and configure OAuth app, [1] for login with Google.

IV. RESULTS

We have developed two different encryption techniques first does encryption using AES and second new algorithm perform encryption using OAuth token we called as Real-time encryption algorithm. The MapReduce programs (Hadoop job) which take the input as encrypted data and execute job, We have taken five consecutive readings for every file size and the average time is recorded in below table. Where as the values of graph are one of the five consecutive readings , from which we have taken average values in the table. We can observe that 101.94 seconds was taken for running a WordCount MapReduce job for encrypted HDFS with AES for size of 10MB test file while 56.18 seconds for the encrypted HDFS with Real-time encryption algorithm(RTEA).

Table 1 : Comparison Between Advance Encryption Standard & Real Time Encryption Algorithm

Data (MB)	Encryption Type	Encrypted Data (MB)	Time Consume For Encryption (Sec)	Time Consume To Upload To HDFS(Sec)
1	AES	1.85	11.48	1.34
	RTEA	1.05	5.65	1.25
2	AES	3.7	22.86	1.41
	RTEA	2.1	12.26	1.31
6	AES	11.1	62.42	1.49
	RTEA	6.3	33.4	1.37
10	AES	18.5	101.94	1.41
	RTEA	10.5	56.18	1.34

Table 1 shows the file encryption comparison between Advance Encryption Standard & new algorithm i. e, Real Time Encryption Algorithm. The result of data upload of plain file & encrypted file shown in the following figures in terms of graphs. The job execution comparison between AES encryption & the new algorithm is shown in Table 2. The results are shown in following figures in terms of graphs.

Table 2 : Comparison Between AES Encrypted Data & Real Time Encryption Algorithm

Data (MB)	Encryption Type	Encrypted Data(MB)	Time Consume For job Execution(Sec)
1	AES	1.85	1.36
	RTEA	1.05	1.30
2	AES	3.7	1.40
	RTEA	2.1	1.38
6	AES	11.1	1.47
	RTEA	6.3	1.42
10	AES	18.5	1.35
	RTEA	10.5	1.31

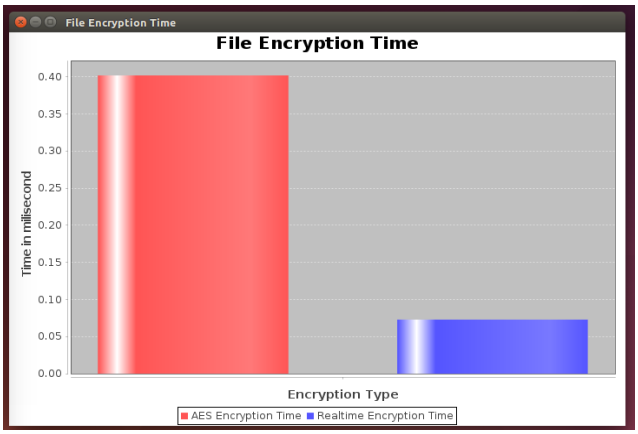

Figure 2: Shows graph of time required to encrypt input file using AES and Real Time encryption algorithm

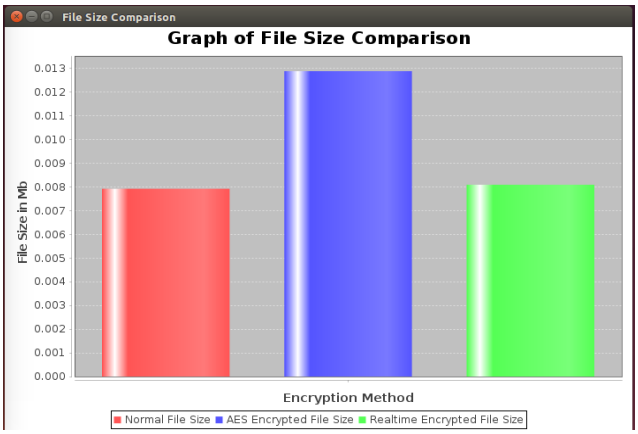

Figure 3: Shows graph of comparison of original file size and file size encryption using AES and Real Time encryption algorithm

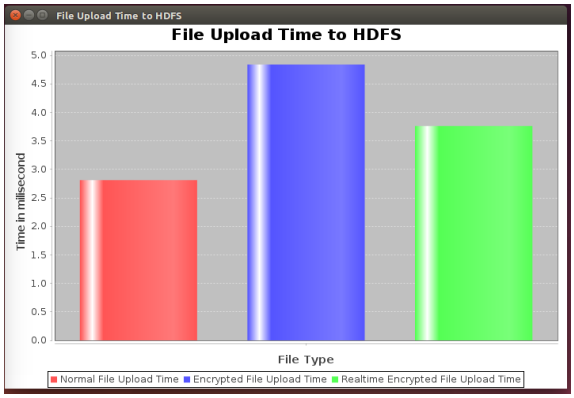

Figure 4: Shows graph of comparison of file upload time of original file and files after encryption using AES and Real Time encryption algorithm

Figure 5: Shows graph of comparison of job execution time of original file and files after encryption using AES and Real Time encryption algorithm

V. FUTURE SCOPE

Big data contains sensitive and private information, in order to protect this big volume that stored at different commodity hardware, necessary to implement authentication to verify user or system identity. Authorization is useful for providing access control privileges to user or system. OAuth 2.0 is good choice for both authentication and Authorization. OAuth 2.0 token powerful mechanism that support AES to provide data confidentiality and integrity among different user.

VI. CONCLUSION

In this time of Big Data, where data is collected from different sources, security is a measure issue, as there is no any fixed source of data and not any kind of security mechanism. Hadoop adopted by various industries to process such data, demands strong security solution. Thus authentication, authorization and encryption or decryption methods are much helpful to secure Hadoop file system.

ACKNOWLEDGEMENT

We would like to thank Shivaji University Kolhapur for giving such wonderful platform for the PG students to publish their research work. Also would like to thanks to our guide & respected teachers for their constant support & motivation for us. Our sincere thanks to Annasaheb Dange College of Engineering & Technology for providing a strong platform to develop our skill & capability.

REFERENCES

[1] S. Ghemawat, H.Gobioff and S. Leung, "The Google File System". In:ACM Symposium onOperating Systems Principles(Octomber 2003).

[2] Thanh Cuong Nguyen, WenfengShen, Jiwei Jiang and WeiminXu, "A Novel Data Encryption in HDFS".IEEE International Conference on Green Computing And Communication 2013.

[3] Raj R. Parmar, Sudipta Roy, Debnath Bhattacharya, Samir Kumar Bandyopadhyay, (Senior Member, IEEE), And Tai-Hoon Kim, "Large-Scale Encryption in the Hadoop Environment: Challenges and Solutions"(2017), DOI 10.1109/ACCESS.2017.2700228.

[4] Marwan Darwish, AbdelkaderOuda, "Evaluation of an OAuth 2.0 Protocol Implementationfor Web ServerApplications"(2015), 978-4799-6908-1/15/$31.00©2015 IEEE.

[5] Seonyoung Park and Youngseok Lee, "Secure Hadoop with Encrypted HDFS", Springer-Verlag Berlin Heildelberg in 2013.

[6] Jason Cohen and Dr.SubatraAcharya, "Towards a Trusted Hadoop Storage Platform:Design Considerations of an AES Based Encryption Scheme with TPM Rooted Key Protections". IEEE 10th International Conference on Ubiquitous Intelligence & Computing in 2013.

[7] Lin H., Shen S., TzengW., Lin B.P., "Toward Data Confidentiality via Integrating Hybrid Encryption Schemes and Hadoop Distributed FileSystem". 26th IEEE International Conference on Advanced Information Networking And Applications in 2012.

[8] Monika Kumari, Dr.SanjayTyagi, "A Three Layered Security Model for Data Management in Hadoop Environment"(2014).

[9] S. Ghemawat and J. Dean, "MapReduce: Simplified data processing on large clusters", ACMCommun. Mag., vol. 51, no. 1, pp. 107_113,Jan. 2008.

[10] D. Borthakur, "The Hadoop distributed file system: Architecture and design",Hadoop Project Website,Aug. 2007.

[11] https://console.developers.google.com

[12] https://developers.facebook.com/apps