International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-2, December, 2019

SecHDFS: Efficient and Secure Data Storage Model over HDFS using RC6

K. Vishal Reddy, Jayantrao B. Patil, Ratnadeep R. Deshmukh

Abstract*: *In today's world the data used by various institutions and organizations is increasing and process petabytes of data per hour. Hence big data storage platform called Apache Hadoop is designed to process large amount of data, but it does not guarantee the security of user stored files in Hadoop. In this paper, a secure HDFS is designed for an efficient and secure data storage model. We encrypt and decrypt client data using RC6 symmetric block cipher. In this research work, Hadoop distributed file system (HDFS) is customized using RC6 which provides transparent end-to-end encryption on user's data for read as well as write. Our proposed SecHDFS will mitigate several security attacks such as replay attacks, data node impersonating attacks, and brute-force attacks. The proposed model imparts better results than the inbuilt AES symmetric algorithm.

Keywords: SecHDFS, AES, RC6, and Hadoop Security

I. INTRODUCTION

Hadoop is a very large storage platform which supports petabytes of data. It mainly comprised of HDFS (Hadoop Distributed File System) and MapReduce. HDFS is a very large distributed system that supports 10K nodes, 10PB and 100 million files. Yahoo is a good example which is working on HDFS version that can scale well even for large amounts of data. Moreover, HDFS provides very high aggregate bandwidth, low-cost, scalable storage for any data type and fast processing of large volume of data. In HDFS, stored files will be broken down into number of blocks and each block size is 64MB. MapReduce is a distributed programming model for efficient computing and keep access data files in a distributed manner. It is best fit for lot of real world applications like log processing and web index building. It becomes very popular for big data processing. However, privacy and security are the primary concerns of the Hadoop Distributed File Systems due to high 3V's (volume, velocity and variety) of big data. A given HDFS client may consists of significant sensitive information. In public health and remote patient data analysis applications, data security is necessary to protect the information. Thus attention towards information leakage is important in the large storage systems. Data encryption is one of the ways to protect sensitive information, which avoids information leakage [1], [2], [3] and [4]. A typical architecture for HDFS against some attackers is depicted in Fig.1

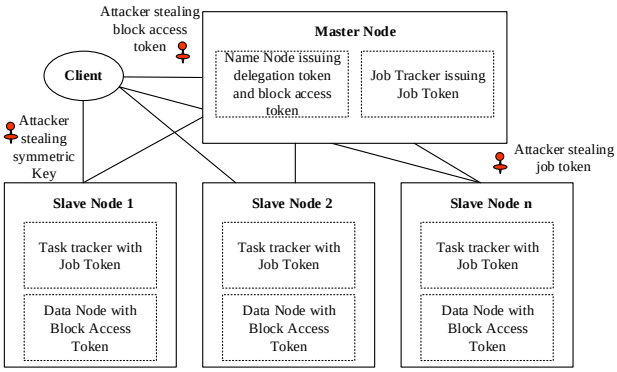

Fig.1. General HDFS Architecture (in a form of Attackers)

A dataset may consist of combination of sensitive and non-sensitive information so that there is a possibility that the HDFS clients who are given access to such data will abuse or misuse it. In specific, HDFS has large size of data gradually increase their sensitive information that requires to be protected. Hadoop uses a third party authentication protocol (Kerberos) to determine authorized users which built on symmetric key cryptography, but it is vulnerable to disguise and replay attacks.

Currently, Hadoop is provided by various cloud providers such as Amazon's Elastic Map-Reduce (EMR), Google Cloud, Horton Works, Microsoft Azure, and Cloudera where importance of security grows rapidly with the increase of Big Data processing in business applications. Some of the primary applications and its categories are described in [5].

II. RELATED WORK

There is several research studies have concentrated to ensure security in Hadoop. Julio et al. [6] proposed Fast-Sec approach to secure big data processing which handles large volumes of data from heterogeneous sources. It is related to the user privacy information protection to address the issues related to authentication and access control mechanism. The encryption server holds all the certified authentication (CA) and encryption operations and grants the user access to the data storage. It stores the encryption keys and connects to a separate key storage server. The files encrypted with 128/256-bit AES and file key (AES unique key) is encrypted for each user. Moreover it contributes three operations

including authentication, certificate authentication and encryption. The limitation of Fast-Sec approach is to encryption size is large for a given plain test size, which increases storage space.

Revised Manuscript Received on December 30, 2019.
* Correspondence Author
K. Vishal Reddy, Department of Computer Science and Engineering, Dr. B. A. M. University, Aurangabad, M.S., India.
Prof. Dr. Jayantrao B. Patil, Department of Computer Science and Engineering, R. C. Patel Institute of Technology, Shirpur, MS, India
Prof. Dr. Ratnadeep R. Deshmukh, Department of Computer Science & Information Technology, Dr. BAMU, Aurangabad, MS, India

Record aware partial compression (RAPC) has proposed by Dong et al. [7], which saves storage space and reduces cost of data transmission over the network. Partial compression produces the maximum value out of data and thereby it is not effective in nature. The combination of compression and encryption is emerging in large scale storage systems. Bruno [8] has considered efficient compression and encryption in digital data transmission. Some of the compression techniques have used such as Huffman coding (HC), JPEG, Run Length Encoding (RLE), JPEG2000, Arithmetic coding (AC) and LZW. When compare with cost of compression (compression size), arithmetic coding provides less compressed size for given test set size. Assume that the given test set size is 3229989 bytes. The compressed size for AC is 1215909 bytes, and compressed size of HC, RLE and LZW are 1833144, 2796235, and 1743266, respectively. In the same way, some encryption methods have compared such as DES (Data Encryption Standard), 3DES, AES and RC4. Then they have applied encryption for compressed size of data. For the compressed size 1215992 (result of AC) bytes, encryption results for DES, 3DES, AES, and RC4 are 1215992, 1215992, 1216080, and 1215909, respectively. From the results, we summarize that the AC and RC4 are good fit for compression and encryption, respectively. However, RC4 provides less processing speed. Hong et al. [9] have designed a new framework called FASTEN (FPGA-based Secure System for Big Data Processing). The plaintext data and security keys are not exposed to main memory and secondary storage. AES is proposed for data encryption and key-pair (public and private keys) are generated using RSA algorithm. However, both RSA and AES consume more storage space and also traditional RSA is very slow for key generation. Jiaqi et al. [10] considered a current deployment of MapReduce over cloud environment. For this purpose, a software framework is designed for individual virtual machines to run a MapReduce application. To store user information, database is used in the top of the master node, which is limited in memory space. Yoon-Su et al. [11] have proposed privacy for user's information. Based on the privacy of user information, Big Data services are provided. For that register their index values, results of chaining and hashing of the user-generated random-bit sequence and user XOR security awareness information. K. Vishal Reddy et al. [12] have proposed two cryptographic algorithms AES and Blowfish which clearly shows the comparative results between AES and Blowfish where AES is better in time complexity and Blowfish is performing better in space complexity.

III. PROPOSED SYSTEM ARCHITECTURE

In our proposed work, three constituents are considered namely Clients, Key Management Server and Hadoop Distributed File System (HDFS). HDFS constitutes of two nodes one acts the master node (NameNode) and other as slave node (DataNode). Today, Hadoop comes with many security threats (replay attacks, impersonation attacks, data node impersonating attacks, stolen verifier attacks, brute-forcing attacks and known-plain text attack). So we require to propose an efficient security model to mitigate such sensitive attacks before storing in HDFS.

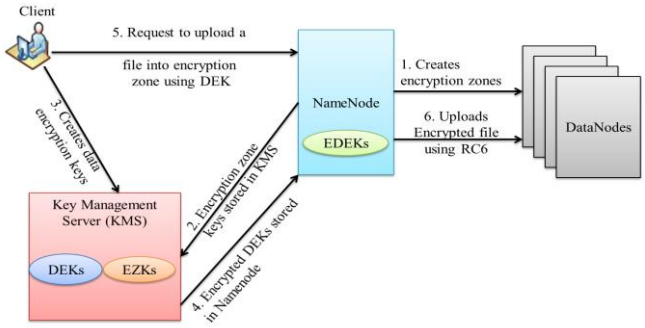

Fig.2. Encryption Process in HDFS

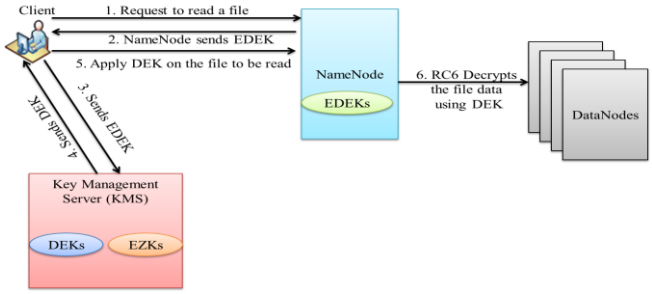

Fig.3. Decryption Process in HDFS

A. System Functioning

The overall encryption process is shown in Fig.2 when client wants to write a file into HDFS.

- Initially super user of HDFS creates an encryption zone using an Encryption Zone Key (EZK).
- These EZK is stored in Key Management Server (KMS).
- Client request the KMS to create new Data Encryption Key (DEK).
- KMS creates DEK and send it to the client simultaneously it encrypts the DEK and the encrypted DEK (EDEK) is stored in NameNode.
- Clients can now upload a file to encryption zone using the DEK.
- By default the HDFS uses AES/CTR/NoPadding encryption with the DEK provided by the client to secure data in the file stored on HDFS.
- In this system, instead of AES/CTR/NoPadding HDFS uses RC6 with DEK provided by the client to encrypt data in the file stored on HDFS.
- The Encrypted File along with EDEK as metadata is persisted in the HDFS.
  The overall decryption process is shown in Fig.3 when client wants to read a file from HDFS.
- The client requests NameNode to read a file.
- In response to the clients request for a file NameNode sends the associated EDEK to the client.
- Client request the KMS to decrypt the EDEK.
- Upon receiving Clients EDEK, KMS checks the file access permission. If access allowed then the corresponding EDEK is decrypted and the DEK is send back to the appropriate client.
- Client uses DEK to decrypt the data in the file.

IV. EXPERIMENTAL SETUP AND RESULT

A. Experimental Environment

The results obtained in this work were carried out on a HDFS testbed configured on 4-core, 2.2 GHz CPU, 1 TB hard disk and 4 GB Memory.

B. Result Analysis

For statistical result analysis we considered the following parameters for AES
W (word size in bits) =16
b (Key length)=255
Block size in bit=256
Number of keys derived=4(r+1)
Used operation = +,-, *, XOR
Parameters for RC6
W (word size in bits) =16
b (Key length)=255
Block size in bit=256
Number of keys derived=2r+4
Used operation = +,-,*, <<, >>, XOR

Table- I: Encryption time analysis with Word Size = 16

Time Consumed		
File Size (in MB)	**AES (in sec)**	**RC6 (in sec)**
105.9	4.77	4.77
213.9	6.19	6.07
303.9	8.89	8.76
402.9	10.48	10.42
501.9	13.294	13.269

Parameters for AES
W (word size in bits) =32
b (Key length)=255
Block size in bit=256
Number of keys derived=4(r+1)
Used operation = +,-, *, XOR
Parameters for RC6
W (word size in bits) =32
b (Key length)=255
Block size in bit=256
Number of keys derived=2r+4
Used operation = +,-,*, <<, >>, XOR

We obtained the results based on performing encryption on plain text files and storing them to HDFS. Encryption in this research work was carried out by using AES and RC6 block ciphers. The results (See Table I and Table II) clearly indicate that, as the file size increases, the encryption time consumed by AES is more when compared to RC6. So, the obtained results show that RC6 is more efficient than the AES algorithm. At the same time, RC6 meets all the requirements of AES. So RC6 can store the data in a more secure manner.

Table- II: Encryption time analysis with Word Size = 32

Time Consumed		
File Size (in MB)	**AES**	**RC6**
105.9	4.44	4.42
213.9	5.89	5.85
303.9	8.36	8.27
402.9	10.38	10.24
501.9	12.948	12.917

The results obtained for space consumption after applying encryption on plain text file is negligible (See Table III).

Table- III: Encrypted File size analysis

Space Consumption after Encryption		
Plain File Size (in MB)	**AES**	**RC6**
105.9	119.13	119.13
213.9	240.36	240.37
303.9	341.89	341.90
402.9	453.27	453.29
501.9	564.65	564.67

V. CONCLUSION

This paper proposes, use of RC6 for providing security for the data at-rest in HDFS. RC6 is simple to implement when compared with AES. The present version of Hadoop-3.2.1 uses AES for securing the data. The proposed use of RC6 instead of AES for Hadoop Framework provides us better results in time complexity. In future we want to extend our work on various block ciphers.

ACKNOWLEDGMENT

This work is supported by Department of Computer Science and Information Technology, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, Maharashtra, India. The authors would like to thank Department and University Authorities for providing the infrastructure and necessary support for carrying out the research.

REFERENCES

1. B. Saraladevi, N. Pazhaniraja, P. Victer Paul, M.S. Saleem Basha, P. Dhavachelvan, “Big Data and Hadoop-A Study in Security Perspective”, Procedia Computer Science, Vol.50, PP. 596-601, 2015
2. Jingxian Xu, Jianhong Guo, Chunlan Ren, “Implementation and performance test of cloud platform based on Hadoop”, IOP Conf. Series: Earth and Environmental Science, Vol. 108, PP. 1-8, 2018
3. Prableen Kaur, Manik Sharma, Mamta Mittal, “Big Data and Machine Learning Based Secure Healthcare Framework ”, Procedia Computer Science, Vol. 132, PP. 1049-1059, 2018.
4. Yannan Ma, Yu Zhou, Chenglei Peng, Ziqiang Wang, Sidan Du, “A Novel Approach for Improving Security and Storage Efficiency on HDFS”, Procedia Computer Science, Vol. 52, PP. 631-635, 2015.
5. Aayush Gupta, Ketan Pandhi, P V Bindu, P Santhi Thilagam, “Role and Access based Data Segregator for Security of Big Data”, Procedia Technology, Vol. 24, PP. 1550-1557, 2016
6. Julio C. S. dos Anjos, Tatiana Galibus, Claudio F. R. Geyer, Gilles Fedak, Joao, Paulo C. L. Costa, Rubem Pereira & Edison Pignaton de Freitas, “Fast-Sec: an approach to secure Big Data processing in the cloud, International Journal of Parallel, Emergent and Distributed Systems”, PP. 1-17, 2017
7. Dapeng Dong, John Herbert, “Content-Aware Partial Compression for Textual Big Data Analysis in Hadoop”, IEEE Transactions on Big Data, PP. 1-14, 2017
8. Bruno Carpentieri, “Efficient Compression and Encryption for Digital Data Transmission”, Security and Communication Networks, PP. 1-9, 2018
9. Boeui Hong, Han-Yee Kim, Minsu Kim, Lei Xu, Weidong Shi, Taeweon Suh, “FASTEN: An FPGA-based Secure System for Big Data Processing”, IEEE Design and Test Hardware Accelerators for Data centers, PP. 1-7, 2017
10. Jiaqi Zhao, Jie Tao, Achim Streit, “Enabling Collaborative MapReduce on the Cloud with a Single-Sign-On Mechanism”, Computing, Vol. 98, Issue. 1-2, PP. 55-72, 2016

11. Yoon-Su Jeong, Seung-Soo Shin, "An Efficient Authentication Scheme to protect User Privacy in Seamless Big Data Services", Wireless Personal Communications, Vol. 86, Issue.1, PP. 7-19, 2016
12. K. Vishal Reddy, Jayantrao B. Patil, Ratnadeep R. Deshmukh, "A Comparative Approach to Secure Data Storage Model in Hadoop Framework", Springer, Advances in Intelligent System and Processing, Vol. 1025, pp. 135 – 144, 2019.

AUTHORS PROFILE

K. Vishal Reddy received the Master of Technology degree in computer science from S. R. M. University, Chennai (TN), India, in 2013. He is currently working towards his Ph.D. degree in the department of Computer Science, Dr. B. A. M. University, Aurabgabad, (MS), India. His research interests include network security, security in Hadoop framework, big data and cloud computing. In particular his research focuses on providing security in Hadoop distributed file system when deployed in public cloud services provided by different vendors. His papers have been published in various conferences and journals such as Springer, IJSEM, IRCSIT, IRIRCCE, IJSR and IJASRET.

Jayantrao Patil received the Bachelor of Engineering in Electronics Engineering from Marathwada University, Aurabgabad (MS), India, M. Tech. Computer Science and Data Processing from IIT Kharagpur, WB, India and the Ph.D. degree in Computer Engineering from N. M. University, Jalgaon, M.S., India. Currently he is the Principal of R. C. Patel Institute of Technology, Shirpur, M.S., India and a university recognized Ph.D. guide. His research interest includes web catching, web mining, text watermarking, web personalization, and web security. He has published more than 50 research papers in renowned journals like IEEE, Springer, Elsewire etc. He got many awards and recognition for his contribution in the field of engineering education. He has been a session chair and key note speaker for various conferences. He has life membership of different international organizations like IEEE, ASEE, ISTE, CSI etc. He received a research grant of Rs. 500000+. He has written a book on "An Integrated Prefetching and Caching Approach – for Web Browsers / Web Proxies". He got a patent for his research on "Method of ROI Extraction for Palm print".

Ratnadeep R. Deshmukh has completed Ph.D. from Dr. B. A. M. University in 2002. He is working as a Professor in Computer Science and Information Technology (CSIT) Department, at Dr. B. A. M. University, Aurangabad (MS) INDIA. He is Executive Committee Member of Computer Society of India, President ICT Section of Indian Science Congress Association (ISCA) and Fellow & Chairman, IETE Aurangabad Center, Life Member ISCA, CSI, ISTE, IEEE, IAEng, CSTA, IDES & SMACEEE. His research interest includes, HCI, Speech Signal Processing, GIS &Remote Sensing, and Data Mining. He has published more than 160 research papers in various National and International Journals and Conferences.