ChaCha20 with Poly a Faster and Secure approach to Encryption of Data in HADOOP

Kaushal D Patel[1] Parth Wadhwa[2]
[1,2]Hasmukh Goswami College of Engineering, India

Abstract— Now, the world is going to become more increase digital. As every person using the internet today. A large amount of data get generated day to day. We are not creating a website in Data mining. We are developing a new approach to secure data. So Data security is an important thing for an organization and unauthorized users. Data security is used for detect the hackers, stolen data, and information. Data security prevents unauthorized users from viewing, updates or modifying the database. Data in map-reduce are non-secured and can be easily captured. So data should be encrypted before sending to a map-reduce algorithm to increase to the security of data. Our approach is to secure data by encrypting it with ChaCha20 with a poly encryption algorithm which is found to be very strong and less time-consuming. It is also found to be better than the current RSA algorithm.

Keywords: Data Security, Encryption, ChaCha20, Decryption

I. INTRODUCTION

Big data is a collection of huge volumes and large amounts of datasets and data volume. Big data include data of social media such as Facebook, email, twitter, you tube, transactional data, large quantities of data, search data. The major characteristics of big data it is also called 3V's (Volume, Variety, and Velocity).

1) Volume: Volume of data refers to the size of data sets is the main characteristic of big data. Big data consist of a very large dataset. It needs to be analysed and processed. An example of the volume of data set would be all credit card transactions in one day.
2) Variety: Variety is in all formats: structured, unstructured, or semi-structured (combination of both). It consists of data sets, it is complicated that traditional data processing applications aren't enough to deal with them. Most examples are structured data -texts, tweets, pictures & videos. An example of the high variety of data sets would be the CCTV audio and video files that are generated at many locations in a city.
3) Velocity: Velocity refers to the speed of data with which data is generated in seconds. High-velocity data is being generated so fast that it requires distributed processing techniques. An example of data that is generated with high velocity would be Social media such as in per second thousands of tweets are tweeted or Facebook posts, send an email, you tube videos.
4) HADOOP: High Availability Distributed Object-Oriented Platform is an open-source framework used for distributed storage and parallel processing on big data. There are two major components HADOOP Distributed File System (HDFS) and Map Reduce.
5) HADOOP Distributed File System (HDFS): It is an important part of distributing data in HADOOP. It is used for distributed storage of data. The input file is first divided into small chunks of the same size except for the last chunk. HDFS have two main components Name node and Data node. The name node is also called the master node. It manages data nodes. Stores Meta data of actual data. Ex. filename, path. The data node is also known as the Slave node. It stored the actual data. The data node is responsible for reading and writes requests for the clients.
6) MapReduce: MapReduce is a main part of HADOOP. Map and reduce task is a combination of both and follows as a MASTER-SLAVE node. It is a programming framework used for processing huge amounts of data. There are two important tasks, Map Task and Reduce Task. The map task is to break the data into other data where individual data tuples are turned into pairs and it takes a converting set. These tuples pairs are key/values. The second task is a Reduce task, which takes the output from the map as a small set of input and combines data tuples. The reduce task is always done after the map task. The function operates on every of the pairs within the input and outputs a unique set of key-value pairs. The output of the map function is then passed to the reduce function as input. The reduce function then, applies associate combination performer on its input, and stores its output to disk. The output of reduce is additionally within the variety of key-value pairs. At the end of reduce, the output is sorted according to the values of keys, and therefore the function for comparison of the key is typically provided by the user. During the execution of a MapReduce job, the input is initial divided into a group of input splits. The map function is applied on every of the splits in parallel. The system spawns one task for every input split, and the output of the task is stored on a disk for transferring it to the reduced tasks. The system starts reduce tasks once all the map tasks are with success completed.
7) Security: Security and their data is a safeguard it is most important in the fields like businesses, organizations, companies, educations, and transportations, etc. When you create a database that store and retrieve data, it is important to protect the data from unauthorized use, disclosure, modification of data. Encryption and Decryption is a main part in cryptography. I used many security in this paper such as AES, RSA, DES, ChaCha-20 and SalSa-20 in cryptography.
8) Advanced Encryption Standard (AES): AES is symmetric block cipher and an encryption algorithm and one of the most popular algorithms in cryptography. Its uses for encryption and decryption processes in cryptography. Advanced Encryption Standard cipher is derived from an aside channel square cipher. Advanced Encryption Standard (AES) is a faster than RSA and Data Encryption Standard (DES) and secure than RSA algorithms. It is strong and secures encryption algorithms for cryptography. AES encryption algorithm is during encryption, the input data as plaintext is divided into 128-

bit blocks. AES contains a number of rounds depending on the length of keys such as 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. We are used 128-bit keys in this paper. After that, there are many operations for the first 9 rounds during encryption. Including operations such as substitute bytes, shift rows, mix columns, add round keys. In the last round number 10 is same operations as the first 9 rounds but without a mix column.

- Advantages: AES algorithm as it is implemented in both hardware and software, it is the most robust security protocol. AES algorithm uses higher length key sizes such as 128, 192, and 256 bits for encryption. Hence it makes the AES algorithm more robust against hacking. AES is a most ordinary security protocol used for wide various applications such as wireless communication, financial transactions, e-business, encrypted data storage, etc. AES algorithm is one of the most spread commercial and open source solutions used all over the world. No one can hack your personal information with the AES algorithm. AES algorithm is makes it very difficult to hack it as a result it is a very safe protocol.
- Disadvantages: AES is uses too simple an algebraic structure. It algorithm is every block is always encrypted in the same way. AES has been hard to implement with software. The AES algorithm in counter mode is complex to implement in software considering both operation and security.

9) Data Encryption Standard (DES): DES is a symmetric block cipher and an encryption algorithm and one of the most popular ciphers in the world. DES uses 64 bits key. DES have used a 64-bit key, but only 56 bits are selected from the 64 bits by the algorithm. In the DES algorithm, the Initial permutation is 64 bit. After that, the input data is divided into two 32-bit parts: the left half and the right half 56 bits are selected from the 64-bit key from PC1. After they are divided into two 28-bit parts. Sixteen rounds of the following operations are then performed: Both halves of the key are rotated left by one or two bits. Then 48 sub key bits are selected by compression permutation. The right half of the data has been expanded to 48 bits by the Expansion Permutation. After that, The XOR operation has been between expanded 48 bits of data and the compressed 48 sub keys. After that, Data have been divided into eight 6-bit pieces from the combined data. Each part is then input to one of the S-Boxes. In the S-boxes, the first 6-bit part is the input of the first S-Box, the second 6-bit is the input of the second S-Box. After that, in 6-bit the first and the last bits are for the row and the rest of the bits for the column of the S-box table. The values in the table where the intersections are found and the values that are found are read and converted to binary format. So the output is 32-bit long of all S-Boxes. The structure of every S-box is different. The output bits from S-Boxes are combined, and they pass from P-Box permutation then, Changed right side bits are added to left side bits. The modified left part of the data becomes a new right part, and the previous right part becomes a new left side. After that, all sixteen rounds, the left and the right parts of the data are combined by the XOR operation. The Final permutation is performed and 64 bit cipher text is generated.

10) CHACHA-20: ChaCha-20 is a variant of SalSa-20 and widely used as an alternative to AES encryption algorithm. It is the most secure lightweight and faster cryptographic algorithm. It is used for fast encryption and decryption in cryptography.

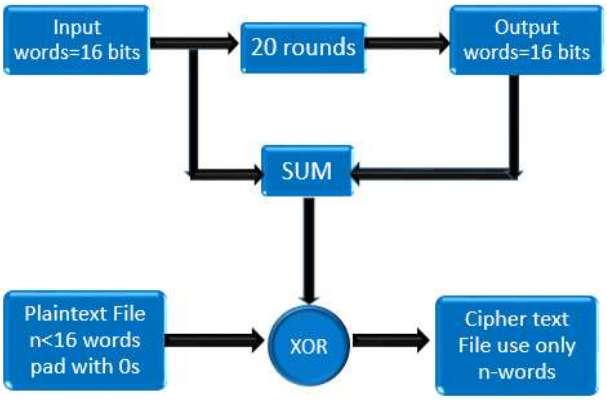


Fig. 1: CHACHA20

Chacha-20 has 4 additions, 4 XORs, and 4 rotations in their operation. It includes a secret key, nonce, and block number. It takes a key (k) of length 32 bytes and a nonce (r) of length 8 bytes as input along with the plaintext. Nonce is a unique value that is never going to changes long as the key is fixed. The pair of (k, r) is never used more than once. So we can reuse the key because the pair of (k, r) is unique. The pair (k, r) is then extended into a 270-byte stream. It encryption a key and plaintext by XOR and generated the cipher text. Decryption operation is the same as encryption as a cipher text is XOR with key and generated the plaintext. ChaCha20 runs 20 rounds between even rounds and odd rounds. The even rounds is a diagonal rounds and odd rounds is a column rounds. The quarter round of ChaCha20 is applied as follows. The algorithm updates four 32-bit input words a, b, c, d as follows:

$$a \mathrel{+}= b;\ d \mathrel{\hat{}}= a;\ d \lll= 16;$$
$$c \mathrel{+}= d;\ b \mathrel{\hat{}}= c;\ b \lll= 12;$$
$$a \mathrel{+}= b;\ d \mathrel{\hat{}}= a;\ d \lll= 8;$$
$$c \mathrel{+}= d;\ b \mathrel{\hat{}}= c;\ b \lll= 7;$$

Where "+" denotes integer addition modulo 2^32, "^" denotes a bitwise Exclusive OR (XOR), and "<<< " denotes an n-bit circular left shift.

- Advantages: ChaCha20 is a faster and speedy algorithm more than RSA, AES, and DES. ChaCha20 usually offers better performance than the Advanced Encryption Standard algorithm ChaCha20 algorithm is a most secured more than RSA, AES and secures up to some size of data. ChaCha20 is a low memory requirement that allows a device to be implemented as small and low-cost. ChaCha20 is relatively ordinary in cryptography libraries with an optimized implementation available for many architectures. ChaCha20 algorithm is the better diffusion and therefore better security.

Cha-Cha following many algorithms.

1) ALGORITHM FOR CHACHA:

Input: K, C, and N
Output: Keystream Z
Generate initial matrix X using Key, Counter, and Nonce
y ← X
For int i ← 0 to 9 do
/* Column Round */
(x0, x4, x8, x12) ←quarter round (x0, x4, x8, x12)
(x5, x9, x13, x1) ←quarter round (x5, x9, x13, x1)
(x10, x14, x2, x6) ←quarter round (x10, x14, x2, x6)
(x15, x3, x7, x11) ←quarter round (x15, x3, x7, x11)
/* Diagonal Round */
(x0, x5,10, x15) ←quarter round (x0, x5, x10, x15)
(x1, x6, x11, x12) ←quarter round (x1, x6, x11, x12)
(x2, x7, x8, x13) ←quarter round (x2, x7, x8, x13)
(x3, x4, x9, x14) ←quarter round (x3, x4, x9, x14)
end for
Z ← X + y
return Z

2) ALGORITHM FOR POLY-1305

Input: Key K and Message M
Output: Tag T

$$(m[0], m[1], \dots\dots\dots, m[d-1]) \xleftarrow{16} M$$

$$d \leftarrow [len\ (M)/16](r, s) \xleftarrow{16} K$$

r ← r&0x0FFFFFFC0FFFFFFC0FFFFFFC0FFFFFFF
For int i ← 0 to d − 1 do
m[i] ← m[i] + $2^{8len(m[i])}$
end for
T ← m[0]
For int i ← 1 to d − 1 do
T ← (r · T + m[i]) mod (2^{130} − 5)
end for
T ← (T + s) mod 2^{128}
return T

3) ALGORITHM FOR CHACHA20-POLY1305 AEAD:

Input: K, N, Authentication data A and Message M
Output: Cipher text C and Tag T
z ← CC-Poly-KS(K, N, len(M))
C ← M ⊕ z
T ← C C-Poly-T (K, N, A, C)
return (C, T)

4) ALGORITHM FOR CHACHA-POLY KEYSTREAM:

Input: K, N and length L
Output: Keystream Z
b ← [L/64]
For int i ← 0 to b − 1 do
z[i] ← ChaCha (K, i + 1, N)
end for
z ← ∑_(i=0)^(b-1)▒ 〖z[i]〗 "• 2512i" ∑
Z ← trancate (l, z)
return Z

5) ALGORITHM FOR CHACHA-POLY-TAG:

Input: K, N, Authentication data A, and Message M
Output: Tag T
k ← trancate(32, ChaCha(K, 0, N))
y ← A
y ← y + M • 2128[len(A)/16]
y ← y + len(A) • 2128([len(A)/16] + [len(M)/16])
y ← y + len(M) • 2128([len(A)/16] + [len(M)/16] + 1/4)
T ← Poly1305 (k, y)
return T

11) SalSa-20: SalSa-20 is secure encryption cryptographic algorithm. Their encryption and decryption process same as chacha20 algorithm in cryptography. SalSa20 runs 20 rounds between even rounds and odd rounds. The even rounds is a row rounds and odd rounds is a column rounds. The quarter round of SalSa20 is applied as follows. The algorithm updates four 32-bit input words a, b, c, d as follows:

b ^= (a + d) <<< 7;
c ^= (b + a) <<< 9;
d ^= (c + b) <<< 13;
a ^= (d + c) <<< 18;

Where "+" denotes integer addition modulo 2^32, "^" denotes a bitwise Exclusive OR (XOR), and "<<< " denotes an n-bit circular left shift.

– Advantages: Salsa20 is an older algorithm in cryptography and therefore better studied more than other algorithm. Salsa20 algorithm is a most secured more than RSA, AES. Salsa20 have been received analysis before its inclusion in the stream portfolio. Salsa20 is not broken, then do not fix it.
– Disadvantages: Salsa20 is obsoleted by ChaCha20. Salsa20 has limited library support. Worse diffusion and confusion for Salsa20.

12) Poly1305: Poly1305 is an authenticator and high-speed message authentication code (MAC). It can be used to test the data integrity and the authenticity of a message. The Final Permutation is performed. The Poly1305 real name is Poly1305-AES message-authentication code, needs a 128-bit AES key.

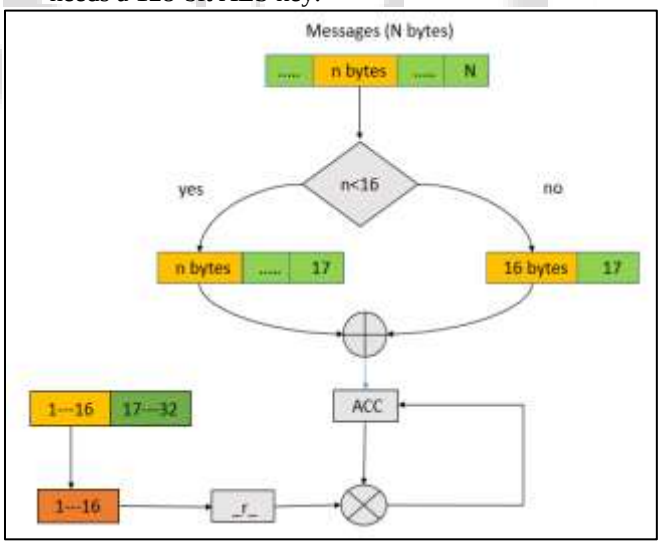


Fig. 2(a): Poly 1305

Poly1305 uses a 32-byte key and operates on an n byte message operation. The first 16 bytes of the one-time key are interpreted as a number r following as. The top four bits of the bytes are cleared, 3, 7, 11, and 15 on indexes. The bottom 2 bits of the bytes are cleared 4, 8, and 12 on indexes, and the 16 bytes are taken as a little-endian value. The accumulator is set to 0. For every n bytes read from the N byte messages. If n = 16 then just add a 17th byte having a value of 1 and the 17 bytes are treated like a little-endian number. If n < 16 then pad with 0s until there are 16 bytes and while n = 16 bytes then add 17. The number is then

added to the accumulator which is multiplied by _r_ and the result is saved back to the accumulator.

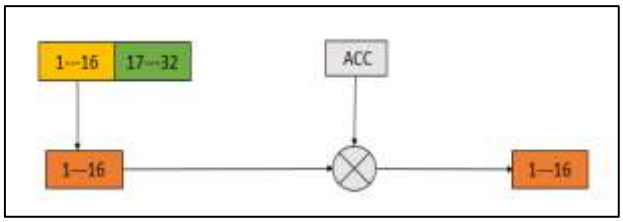


Fig. 2(b): Poly 1305

In lastly, the last 16 bytes of the key interpreting as a little-endian number. And this number is added to the accumulator mod 2^128. The result is then written out as a little-endian number and this is taken as the 16-byte tags.

II. PROPOSED ALGORITHM

In this paper, we introduced CHA-CHA 20 with POLY 1305. Which is a member of the SALSA 20 e-stream family for encryption purposes. Stream cipher takes plaintext and produces ciphertext by combining pseudo-randomly generated keystream. In-stream cipher, Returns the ciphertext by bitwise operation of plain text with the corresponding digitChaCha-20 is a variant of SalSa-20 and widely used as an alternative to AES encryption algorithm. It is the most secure lightweight and faster cryptographic algorithm. It is used for fast encryption and decryption in cryptography. Chacha-20 has 4 additions, 4 XORs, and 4 rotations in their operation. It includes a secret key, nonce, and block number. It takes a key (k) of length 32 bytes and a nonce (r) of length 8 bytes as input along with the plaintext. Nonce is a unique value that is never going to changes long as the key is fixed. It encryption a key and plaintext by XOR and generated the cipher text. Decryption operation is the same as encryption as a cipher text is XOR with key and generated the plaintext. ChaCha20 runs 20 rounds between even rounds and odd rounds. The even rounds is a diagonal rounds and odd rounds is a column rounds.

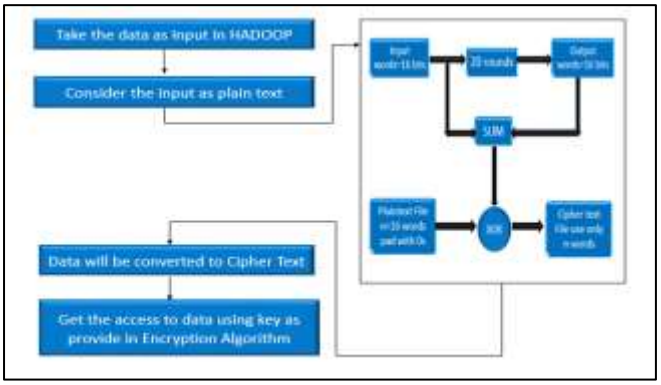


Fig. 3(a): Encryption on Proposed Methodology

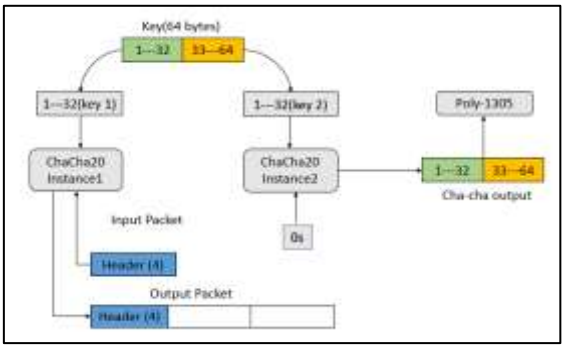


Fig. 3(b): ChaCha20-Poly1305

ChaCha20-Poly1305 is an Authenticated Encryption mechanism that combination of two algorithms. Such that ChaCha20 for Encryption and Poly1305 for Authentication. ChaCha20-Poly1305 uses a 64-byte symmetric key. The input packet contains a 4-byte header packet length, as well as variable-length payload encoding.

Packet Length = Length of Header + Length of Payload + Length of MAC.

The first is ChaCha20 is used to be encryption of the Header using the first 32 bytes of the key and an Initial vector as follows:

Block Counter = 0s

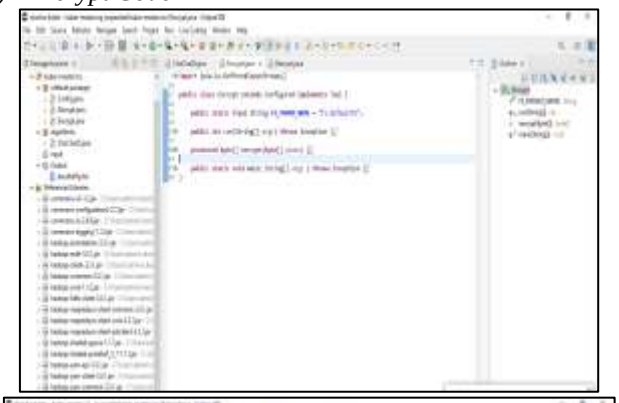
Nonce = Packet Sequence Number (PSN

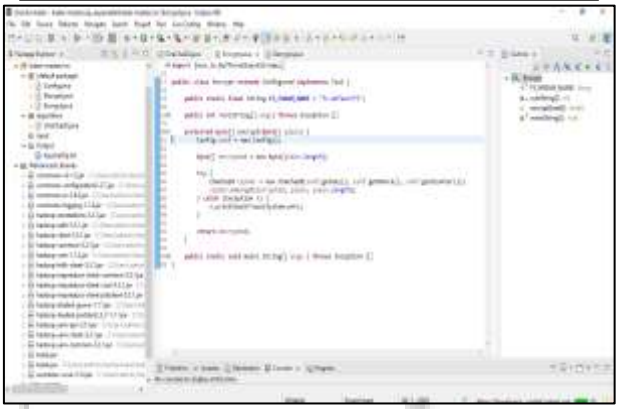
A second is used to be generating of a key for Poly1305 by using the last 32 bytes of the key, 0s as input, and keep the first 32 bytes of output. This second illustration is then used with Block Counter = 1 in Little Endian (LE) and Nonce = PSN to encrypt the n-byte payload. The ChaCha20 will enhancement in Block Counter internally but the ChaCha20-Poly1305 implementation should be managing the Nonce. Finally, Poly1305 is used on the encrypted header and payload and connected to the output packet by calculating one MAC. For decryption, decrypt the header to get the length then check the MAC and decrypt the remaining packets only if the MAC is valid.

III. IMPLEMENTATION AND RESULT

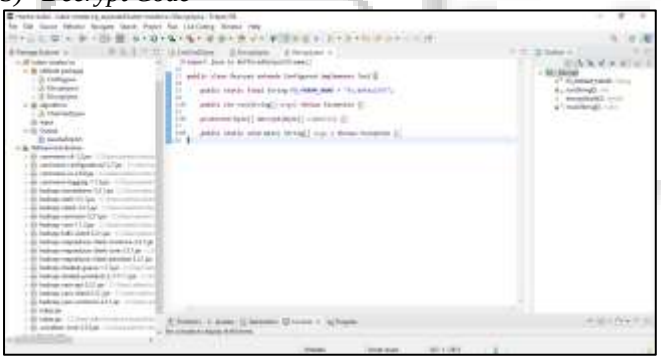
1) Project File and Folder Structure

2) Encrypt Code

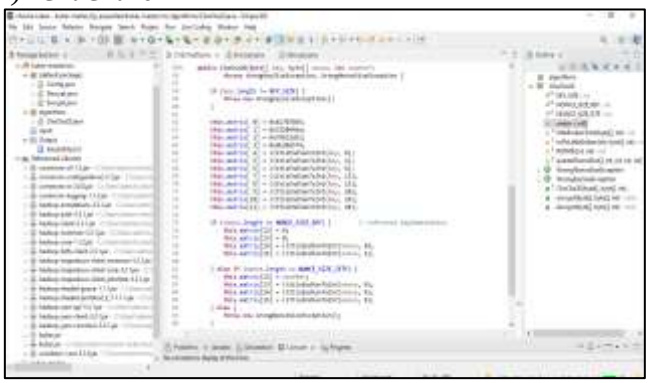




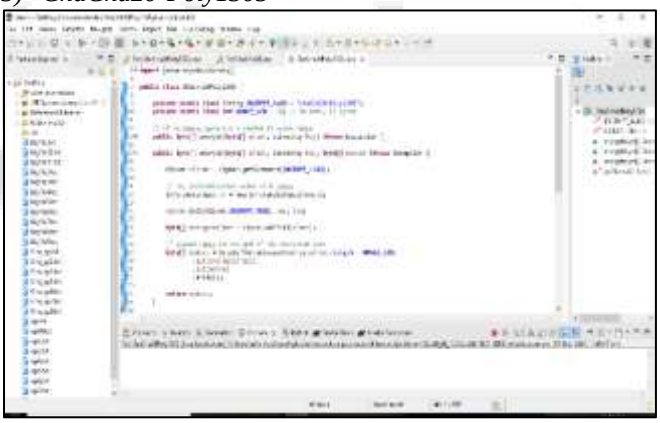
3) Decrypt Code



4) ChaCha20

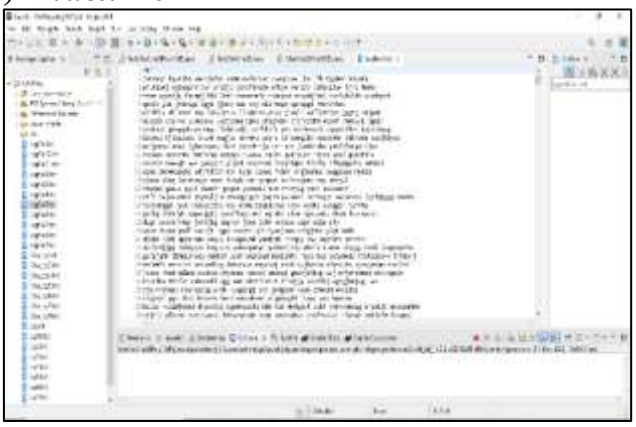


5) ChaCha20-Poly1305



6) Test File

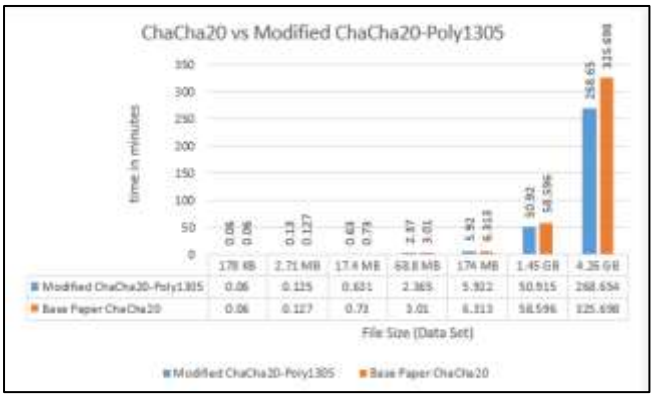
7) Dataset File



8) Output File

IV. RESULT

	Data Set File name	Proposed Algorithm Modified ChaCha20-Poly1305	Base Paper Cha-Cha
178 KB	Bigfile2	0.06 minutes	0.06 minutes
2.71 MB	Bigfile4	0.125 minutes	0.127 minutes
17.4 MB	Bigfile3	0.631 minutes	0.730 minutes
68.8 MB	Bigfile7	2.365 minutes	3.01 minutes
174 MB	Bigfile	5.922 minutes	6.313 minutes
1.45 GB	Bigfile9	50.915 minutes	58.596 minutes
4.26 GB	bigfile11	268.654 minutes	325.698 minutes



V. CONCLUSION

In this paper, we discussed about a cryptographic method to secure data in the HADOOP environment. On experimenting our proposed approach on certain large dataset it is found that it performs way better than current algorithm. Comparison has been made on parameters like time to encrypt the data, Accuracy, Integrity and Security.

We will use Cha-Cha20 with poly to improve the system and provide good results. This algorithm provides better security to data in the HADOOP system using Cha-Cha20 with poly algorithms. Finally, thus our proposed algorithm using Cha-Cha20 with Poly will provide more security to the HADOOP system. Also running time of the proposed algorithm is less compared to the existing algorithms. The encryption and decryption process is faster with additional security at no extra cost.

REFERENCES

[1] Parmar, R. R., Roy, S., Bhattacharyya, D., Bandyopadhyay, S. K., & Kim, T.-H. (2017). Large-Scale Encryption in the Hadoop Environment: Challenges and Solutions. IEEE Access, 5, 7156–7163.
[2] Apache Hadoop, https://en.wikipedia.org/wiki/Apache_Hadoop#History, accessed on December 2016.
[3] Apache Hadoop 2.7.2 - Transparent Encryption in HDFS, https://hadoop.apache.org/docs/r2.7.2/hadoop-project-list/hadoophdfs/TransparentEncryption.html, accessed on February 2017.
[4] HDFS Data at Rest Encryption, https://www.cloudera.com/documentation/enterprise/5-4- x/topics/cdh_sg_hdfs_encryption.html, accessed on February 2017.
[5] HDFS Encryption Overview, https://docs.hortonworks.com/HDPDocuments/HDP2/HDP2.3.2/bk_hdfs_admin_tools/content/hdfs-encryption-overview.html, accessed on February 2017.
[6] Advanced Encryption Standard – Wikipedia, https://en.wikipedia.org/wiki/Advanced_Encryption_Standard, accessed on January 2017.
[7] Salsa20 – Wikipedia, https://en.wikipedia.org/wiki/Salsa20, accessed on February 2017.
[8] Zhu, N., Liu, X., Liu, J., & Hua, Y. (2014). Towards a cost-efficient MapReduce: Mitigating power peaks for Hadoop clusters. Tsinghua Science and Technology,
[9] Hdfs. Encyclopedia of Big Data Technologies.
[10] Cheng, D., Zhou, X., Lama, P., Wu, J., & Jiang, C. (2017). CrossPlatform Resource Scheduling for Spark and MapReduce on YARN. IEEE Transactions on Computers
[11] A. Katal, M. Wazid and R. H. Goudar, "Big data: Issues, challenges, tools and Good practices," *2013 Sixth International Conference on Contemporary Computing (IC3)*, 2013, pp. 404-409, doi: 10.1109/IC3.2013.6612229.
[12] Menon, S.P.; Hegde, N.P. "A survey of tools and applications in big data "Intelligent Systems and Control (ISCO), 2015 IEEE 9th International Conference, pp: 1 - 7.
[13] Apache Hadoop, http://hadoop.apache.org/
[14] https://www.edureka.co/blog/mapreduce-tutorial

