IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727
PP 29-34
www.iosrjournals.org

Secure File Handling on Cloud based on Hadoop using HDFS

Tambe Rohini[1], Pansare Tejashri[2], Hadawale Megha[3], Bhor Pooja[4],
Prof. Salunkhe T. R.[5]
[1,2,3,4,5]*(Department of Computer Engineering, SVCET, Rajuri/Savitribai Phule Pune University, India)*

Abstract: *Hadoop is an Apache open-source framework for storing and processing large amount of data across clusters of computers. But processing sensitive or personal data in hadoop framework requires security model. As we know that hadoop was designed without any security model, in this paper we present self-destructing data system that meets this challenge through a novel integration of secure cryptography techniques with active storage techniques based on hadoop. In this system, we present Shamir's secret sharing algorithm against sniffing attacks by using the public key cryptosystem to protect from sniffing operations.*
Keywords: *Cloud Computing, Hadoop, MapReduce, HDFS, Self Destructing Data, Active Storage .*

I. Introduction

In 21st century everyone refers to store data on the Cloud .That data may contain account numbers, passwords and other important information that could be used and misused by a criminal or an intruder. This data is retrieved, copied and achieved by Cloud Service Providers (CSPs), often without user's permission and control. These problem presents challenge to protect people's privacy from illegal actions. By taking this problem into consideration, for this here we introduce self-destructing system to protect people privacy based on active storage framework.

In proposed system, we present self-destructing data system that meets this challenge through a novel integration of secure cryptography techniques with active storage techniques based on hadoop. We implement a proof-of-concept SeDas prototype. By using functionality and security properties the SeDas prototype can be evaluated. SeDas is practically easy to use and achieve all the privacy preserving aims described above. Compared to the system without self-destructing data mechanism, performance of uploading and downloading acceptably decreases, while latency for uploading and downloading operations with self-destructing data mechanism increase.

II. Literature review

Let us see some theoretical and methodological contributions to a particular topic. *Ling fang Zeng*, proposed improved Washington's Vanish system for self-destructing data under cloud computing, and it is open to "hopping attack" and "sniffer attack". In this paper working of Safe Vanish, to prevent hopping attacks by way of Increasing the length of the key shares to rise the attack cost did some more enhancement on the Shamir Secret Sharing algorithm implemented in the Original Vanish system. They present an improved approach to prevent sniffing attacks by using the public key cryptography system to protect from sniffing operations. In addition, they evaluate analytically the functionality of the proposed Safe Vanish system[5].

Yu Zhang, Introduced that paper we present a reconfigurable calculating solution that can provide high-performance, flexible processing capabilities for the storage nodes. The dynamic reconfiguration upturns the functional density; however, the configuration self results in extra overhead, which may make the overall performance be downgraded. In the future works, we will implement multiple Processing Elements in the reconfigurable accelerator, and design an efficient method for dispatching the Processing Elements to hide the reconfiguration latency to improve the performance[11].

FU Xiao, Realized emails were being watched by the government. For the advantage that Big Data technologies such as large distributed storage and user behavior analysis and so on emails became one of the highly popular Big Data that has been targeted at as a large source of intelligence by some organizations keeps eye on public accounts every hour every day. research work was just opposite to what the NSA has did: To design and implement a system which can store emails securely, and terminate them clearly when they expired. In another word, a self-destructing emails system. But in this system there is no parallel processing for multiuser access[3]. In the existing system there are multiple disadvantages are available. In this Hacker can attack the confidential data and gain all the information from the database. This is big disadvantages of this system. Because client want to security of the data which is confidential from other's. In this hacking process the sensitive data can be modified by anyone, or if anyone can do changes in this client data.

III. Problem Statement

The most often problem while using Cloud and mobile computing is security of personal data stored on the cloud and handling the multiple client node efficiently without affecting the speed of data transferring from server. In case of security, one will always prefer to cloud for storing his personal data. That data may contain account numbers, passwords and other personal information. The personal information may get misused by intruder, dark side hackers, etc. While handling multiple client, the server may slow down and results into less throughput. So main motive is to handle multiple client along with maximum throughput.

IV. Goals And Objectives

- Improve the efficiency of server while handling multiple clients.
- Improve the security while transferring the secrete data.
- Use effective partitioning technique to minimize time complexity of algorithm.

V. System Architecture

Initially, the client has to register at metadata server. After registration, client has to perform login operation. For performing operations, valid user has to enter into database with session. At the metadata server, MapReduce framework accepts multiple client request to register them on server. In which, clients requests are divided by MapReduce to decrease the load of server. To check the validation of user, divided part of session key for each client will be forward to client as well as to the storage node. To validate user, there is need to conquer this parts of session keys at storage node and metadata server. If entered user is valid then metadata server provides access to the database for file operations such as uploading and downloading. As we using Shamir's algorithm, security is also increased.

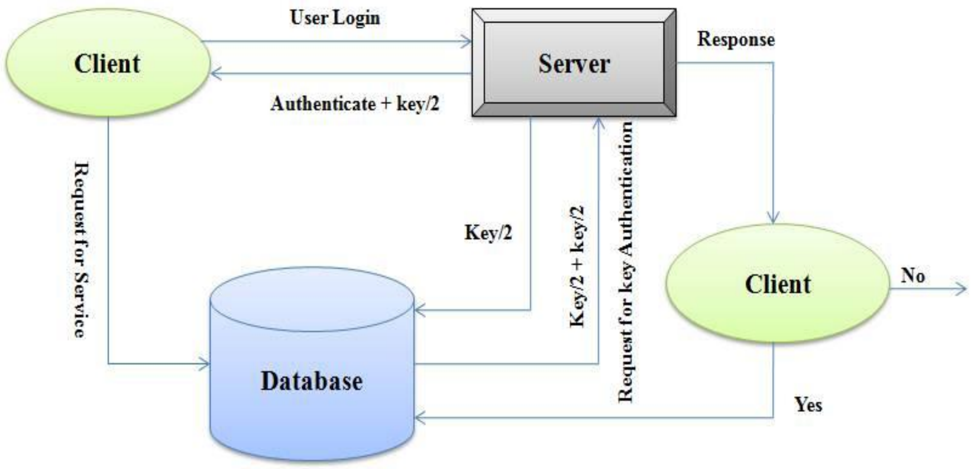

Fig 1. system architecture

5.1 MODULES USED IN PROPOSED SYSTEM

Our proposed system is divided into following modules:

1. Registration: In registration phase, we are taking the user details. If user was registered already by using attributes specified in the registration phase, then that user is discarded from registration. If user was not registered, then the user registration is processed and database is updated with generation of secrete key.
2. Login: In login phase, the user login details are taken from user. After taking user login details we are checking for user valid or not in our database.
3. Split: When user enters key, then this key is divided into 'n' shares from which one share is given to client and another is given to databases.
4. Encrypt: Before uploading file, it is convert from plaintext to cipher text using public key cryptography technique.
5. Upload: Here, user download the encrypted file.
6. Combine: As the key shares are distributed among all storage nodes and one share is distributed to client. To authenticate any file operation there is requirement to gather all required shares to reconstruct the key.
7. Decrypt**:** Before downloading file, it is convert from cipher text to plaintext using public key cryptography technique.
8. Download: Here, user download the decrypted file.

5.2 METHODOLOGY USED IN PROPOSED SYSTEM

5.2.1 Hadoop

Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

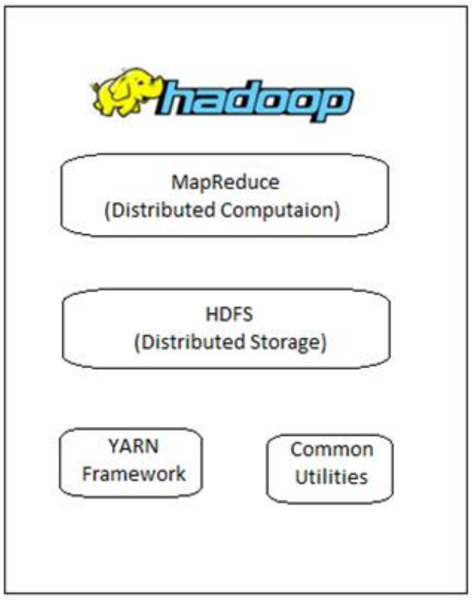

Fig 2. hadoop architecture

At its core, Hadoop has two major layers namely:
(a) Processing/Computation layer (MapReduce)
(b) Storage layer (Hadoop Distributed File System).

5.2.2 MapReduce

From past few years, there is an exponential growth and availability of data, both structured and unstructured. Structured data of traditional database to unstructured data of social networking sites, simple data like text data to complex data like video data are increasing at high rate. More is the data higher is the complexity of analyzing it. MapReduce is a programming framework developed by Google in 2004 for processing of large datasets across distributed systems. Basically, MapReduce is used to simplify data, processing across massive datasets. It is an abstraction to organize parallelizable tasks.
Workflow of MapReduce mainly operates on key: value pairs that is input is submitted as a job of key: value pair and produces set of key: value pairs as a output of job.

Generally three main phases are involved in MapReduce:

I. Map Phase: It take some input data from user and map it to key: value pairs as per specifications provided by the user.
II. Shuffle Phase: At this intermediate stage, key: value pairs emitted from mapper are collected. Pairs with same key are grouped together and passed to reducer for further processing.
III. Reduce Phase: At this phase, intermediate list of key: value pairs are reduced according to user specified reduce function and produce output of multiset key: value pairs with same key are generated. This is a phase where user gets his/her expected output.

5.2.3 HDFS

The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS) and provides a distributed file system that is designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. It is highly fault-tolerant and is designed to be deployed on low-cost hardware. It provides high throughput access to application data and is suitable for applications having large datasets.

Apart from the above-mentioned two core components, Hadoop framework also includes the following two modules:

Hadoop Common: These are Java libraries and utilities required by other Hadoop modules.

Hadoop YARN: This is a framework for job scheduling and cluster resource management.

5.2.4 Shamir's Algorithm

Using Shamir's algorithm we divide our generated session key and share it among the nodes and one share is provided to client. To perform any operation on database client need to provide that share. If all the needed shares must be provided correctly then permission is granted to client to perform operation on database. If any of the needed share is lost then operation has been discarded by metadata server.

5.3 MATHEMATICAL MODEL ASSOCIATED WITH PROPOSED SYSTEM

System S:{ Q, Σ, O, δ, I, DD, NDD}
Where,
Q is set of states.
Σ is input given by user to select state(q)for its respective operation.
O is Output generated by state(Q).
δ is transition function which map $Q*\Sigma \rightarrow Q$.
I is input value x given by user.
F is Functions to system.
DD is Deterministic data.
NDD is Non-deterministic data.
$S:Q(I) \rightarrow Q$
I: Input value given by user.
Where,
Input={ Key │ A - Z, a - z, 0 – 9}
Q :{ $q_0, q_1, q_2, q_3, q_4, q_5$}
Where,
q_0 = Initial state user registration.
q_1 = User Login.
q_2 = File uploading by taking encryption key from user.
q_3= Repeat step1.
q_4= File downloading by taking decryption key.
q_5 = Exit.
Σ: { Ek, eFL, ttl }
Where,
Ek ={ Ek │ A - Z, a - z, 0 – 9}
eFL = { eFL │A - Z, a - z , 0 – 9 }
ttl ={ ttl │ A-Z, a-z, 0 – 9 }
Ek ={ Ek │ Sh1 + Sh2 + . . .+ Shn }
Sh1/Sh2/. . . /Shn = { Sh1/Sh2/ . . . /Shn │ A - Z, a - z, 0 – 9 }
Where,
Ek is Encryption key,
eFL is Encrypted file,
ttl is Time-To-Live,
Sh1,Sh2,...Shn is Key shares.
O:{ dFL, Dk }
Where ,
dFL = { Df │Df(CT:Dk) }
Where,
dFL is decrypted file,
CT is cipher text,
Dk is Decryption key
Dk = {Dk │ A - Z, a - z, 0 – 9}
Dk = {Dk │ Sh1 + Sh2 + . . .+ Shn}
Shn is Share at threshold value.
δ: { Upload File(), Download File(), ttl() }
Upload File (f, s) =P :: takes the file and encryption key.
P = { h │ h takes file, s │s takes encryption key }
Download File (d, k) = A:: outputs decrypted file and takes decryption key.

A = {d │ d outputs decrypted file, k │ k takes decryption key}
ttl (f, t) =B :: f takes encrypted file, t takes time to remove file.
DD: Set of Deterministic state data which gives required output.
NDD: Set of Non-deterministic state data which doesn't gives required output.

5.4 ADVANTAGES OF PROPOSED SYSTEM

- System can balances the load of server.
- System has ability to handle multiple clients.
- It also provides the security to sensitive data while transferring file.

VI. Result

As the Data encryption and decryption is done by using Shamir's algorithm, it becomes more secured. Also there is ttl field ,so that data expires at valid time out.

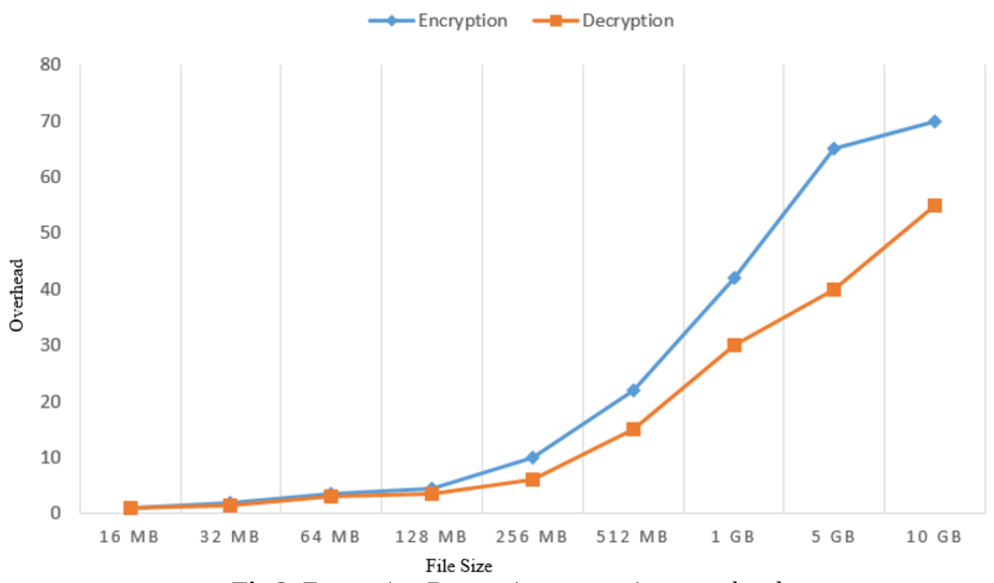

Fig.3. Encryption Decryption comparison overhead

Efficiency of uploading and downloading has been increased as we are using HDFS which works on Map-Reduce. Map-Reduce is works efficiently on large datasets.

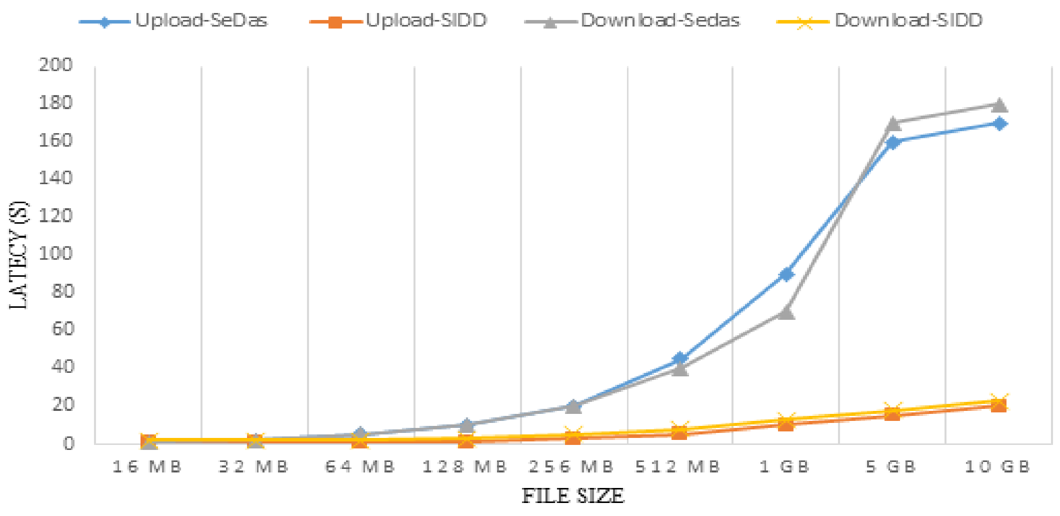

Fig.4. Comparison of Latency in upload and download operation

VII. Conclusion & future work

Hence, proposed system provides security to our personal data with help of hadoop. Hadoop has been efficient solution for companies dealing with the data in Peta bytes. According the above sections we can say that hadoop is one of the best ways to provide the security to sensitive data.

As a future work, we will extend new scheme to deal with other desirable features such as cloud based services and so on. We will also study the applicability of our newly proposed scheme to real world application.

References

[1]. Wenfen Liu, Xuexian Hu, "*Secure Data sharing in Cloud Computing using Revocable Storage Identity Based Encryption* ", IEEE Transactions on Cloud Computing, VOL. *14*, NO. *8*, AUGUST 2015.

[2]. Shaofeng Zou, Student Member IEEE, Yingbin Liang, "*An Information Theoretic Approach to Secret Sharing*", IEEE Transactions On Information Theory, VOL. *61*, NO. *6*, JUNE 2015.

[3]. FU Xiao, WANG Zhi-jian, WU Hao, YANG Jia-qi, WANG Zi-zhao, "How *to send a Self-destructing Email*", IEEE International Congress on Big Data, 978-1-4799-5057-7/14 2014.

[4]. R.C.Dharmik,Hemlata Dakhore,Vaishali Jadhao,"*Sedas: A Self Destructive Active Storage Framework for Data Privacy*" ,International Journal of Scientific Engineering and Research,Volume *2*,Issue *3*,March 2014.

[5]. Ling fang Zeng ,Shibin Chen , Qingsong Wei,"SeDas*:A Self-Destructing Data System Based on Active Storage Framework*", IEEE Transactions On Magnetics , VOL. *49*, NO. *6*, JUNE 2013.

[6]. Mrudula Varade, Vimla Jethani, "*Distributed Metadata Management Scheme in HDFS*", International Journal of Scientific and Research Publications, VOL. *3*, NO. *5*, May 2013.

[7]. Xukai Zou, Fabio Maino, Elisa Bertino,Yan Sui,Kai Wang and Feng Li,"*A New Approach to Weighted Multi-Secret Sharing*", IEEE, 978-1-4577-0638-7/11,2011.

[8]. L. Zeng, Z. Shi, S. Xu, and D. Feng, "*Safevanish: An improved data self-destruction for protecting data privacy*",IEEE,978-0-7695-4302-4/10,2010.

[9]. Cong Wang,Qian Wang,Kui Ren, "*Privacy- Preserving Public Auditing for Data Storage Security in Cloud Computing* ", IEEE,978-1-4244-5837-0/10,2010.

[10]. Seung Woo Son, Samuel Lang, Philip Carns,Robert Ross, Rajeev Thakur,"*Enabling Active Storage on Parallel I/O Software Stacks*", IEEE,978-1-4244-7153-9/10, 2010.

[11]. Yu ZHANG, Dan FENG, "*An Active Storage System for High Performance Computing*", IEEE 22nd International Conference on Advanced Information Networking and Applications,1550-445X/08, 2008.

[12]. Tina Miriam John, Anuradharthi Thiruvenkata Ramani,John A. Chandy,"*Active Storage using Object-Based Devices*",IEEE,978-1-4244-2640-9/08,2008.