ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

Enhanced Chunk Level Deduplication with Security in Hybrid Cloud Data Storage Approach

Sunita S. Velapure[1], Saudagar. S. Barde[2]

M. E. Student, Dept. of Computer Engineering, SKNCOE, Savitribai Phule Pune University, Pune, MH, India[1]

Assistant Professor, Dept. of Computer Engineering, SKNCOE, Savitribai Phule Pune University, Pune, MH, India[2]

ABSTRACT: Data deduplication is an important procedure use for eliminating redundant data files in cloud storage systems. In this, Instead of taking number of same files, it store only single copy of file. In most of the storage systems of many organization contain many duplicate contents in various data files. Like, multiple users save the same files with different files names in different storage locations. Because of this, memory is wasted so we use deduplication concept. By using deduplication, only single copy of data is stored on server and extra copies are not allowed to store on server and these duplicate copies are replace with pointers which points to original copy. This data compression technique is used for efficient bandwidth and memory utilization. For security purpose, data is always encrypted before uploading on cloud servers. In this paper we developed authorized chunk based deduplication checking, combine with encryption technique for providing security over sensitive data using hybrid cloud computing structure, in which users are assigned with some token which is generated at private server side for valid deduplication checking.. Experimental result of proposed system shows that it is more secure and consumes less memory on cloud for storing huge amount of data. We have also shown that proposed scheme has minimum duplicate removal overhead as compared to the existing system with normal deduplication technique, because proposed system makes use of chunking concept.

KEYWORDS: Data duplication; chunking; encryption; confidentiality; hybrid cloud.

I. INTRODUCTION

Cloud computing has wide scope now a days. Cloud provides large amount of virtual environment, hiding the platform and operating systems of the user. User can use the resources as per their demand and pay for the services received on cloud. Now a days, cloud service providers are offering their services with very low price along high reliability. Large amount of data is uploaded on the cloud storage systems and it is also shared by huge amount of the users. Cloud Service Providers offer different kind services such as: infrastructure as a service (IAS), platform as a service (PAS), etc. User does not needs to purchase the resources. As the huge amount of data is uploaded by millions of users every day, it is very critical and challenging task to manage such continuously increasing data on the cloud storage servers. For proper data management in the cloud storage systems, deduplication of the data introduced in [7] is the best method. This method is becomes more attraction point of cloud service providers, now a days. Data duplication is the technique used for reducing the size of data and reduce the wastage of memory on cloud servers or it is the also known as the best compression concept for the data deduplication.

The deduplication methods are widely used in numerous application such as data management and also in the networking, for small amount of unique data routing through network. Instead of storing repeated copies of the same data files, deduplication only allows original copy and provide only pointer references of the original copy to the redundant data file. There are two methods available for duplication checking such as: 1) file level duplication check and other is 2) chunk level duplication check. In the first method, only the file with same name are removed from the storage whereas in second, which is in chunk level deduplication, the duplicate chunks of same files are removed. As the data deduplication is considering with the user data, the need of some strong security technique is arises. Which means that the security and privacy concern of the user’s sensitive data is arises. To avoid the unauthorized data deduplication problem, the solution is proposed in [8] which strictly consider the data confidentiality at the time of data

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

duplication checking. We provide the security over sensitive data by using some strong encryption methods. We encrypt every chunk of file before data duplication checking and only encrypted files are store on cloud servers.

To check the authorized duplicate contents we are using user tokens to check the authentication of the user. In the hybrid cloud the user tokens are present at the private cloud and data of the user is at public cloud. The hybrid cloud structure take advantages of both public as well as private cloud. When any user forward request to the public cloud to upload or download the data he need to submit his information to the private cloud then private cloud will generate and distribute a file token and on receiving this token user can access to the file resides on the public cloud. In the proposed system we have used hybrid cloud architecture. We perform the data duplication based on chunks and hash of chunks of files. If user wants to retrieve data or download the data file he need to request to private server and then public server returns the chunks into single file by using merge operations. The proposed system is secure and has minimum overhead of chunk level data deduplication checking.

II. Related work

In the literature, most of the work has been done on methods of cloud storage and data deduplication techniques. One of the data deduplication method is post processing method [3] in which, first data is stored on the storage device and then duplication checking is applied on that data. The advantage of this method is that there is no need to wait for hash function calculation and the speed of storage is not reduced. The main drawback of this system is that, the file storage might be insufficient on arrival of unmanaged data. In this way, the post processing method is not working at all because it checks the duplicate file contents after uploading on the cloud server.

Another duplication check method is named as inline duplication check [5]. The name is given because it checks duplication on arrival of new entries, after this, the unique data is added to the cloud database server. Before data upload, system performs block level duplication of the file. But this system may leads to slower throughput of the storage device because every time system needs to calculate the hash function. Even though, most of the researchers has proven that the output of inline and post processing data duplication check system has same output.

Next method is known as source duplication check proposed in [6]. In this system, duplicate file contents are checks for before uploading it on the cloud data server.

The source data deduplication is implemented in [7], where the data duplication checking is carried out on user side to reduce the storage and processing overhead at cloud server side. The duplication is checked at the target level where files are scanned periodically and hash is generated. Then the calculated hash values are compared. If the new hash value is matched with the existing hash value then the new file is not allowed to be uploaded on the cloud server. In this case, only link to that similar data is send to the data owner. But here the problem is that whole files is consider for duplication checking. The concept of chunk level duplication is not used here.

The problem identified in [7] is solved in [8]. In this system, chunk level file duplication checking is implemented. The file is divided into chunks of equal size and then for each chunk, duplicate contents are verified.

III. Proposed algorithm

A. System Overview

Figure 1 represent the architectural view of our proposed system. This system is used for data duplication checking before storing the files on cloud server. Here we makes use of hybrid cloud that is both private cloud and public cloud. Both cloud servers perform different levels of responsibilities. User can upload, download or modify the files stored on public server. The data stored on public cloud is confidential therefore to avoid the access of unauthorized users, we makes use of data confidentiality techniques. The authentication of user and security of data is performed at private server. For accessing the data on public cloud, user send request to private cloud. In return, private cloud generate the key pair (public key, private key) for each user by using RSA key generation algorithm. Private server is responsible for key generation and key management. After getting the keys, user encrypt the files to store on public cloud. But before sending the data to public cloud, user needs token for data duplication checking. So user now requesting to private cloud server for token. Private server generates the token by using AES algorithm. The key of user is encrypted by using AES algorithm and this encrypted key treated as a token.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

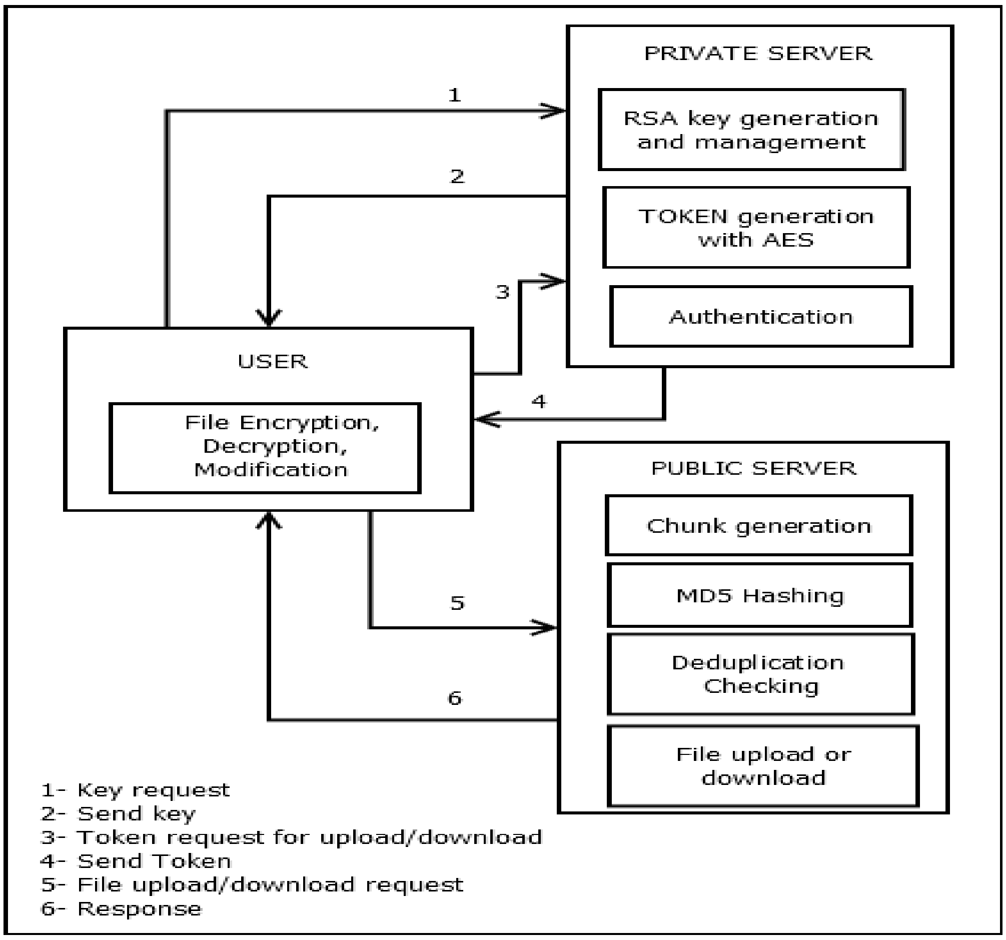

Fig. 1: Data Deduplication checking at Public server.

After receiving token, user send the data to public cloud server with file upload requests. Now before storing the data, this public server performs data deduplication checking to avoid the upload of same data files. This will help to reduce the wastage of memory with duplicate files. In proposed system, we use chunk level deduplication of files at public cloud server.

At receiving the file upload request, initially public cloud creates chunks of received files. For chunking, we divide the file into chunks of 10mb. After this, hash of each chunk is generated by using MD5 hash generation algorithm. In public cloud server, the hash of all chunks of previously stored files are saved. For data deduplication checking, public server compare the hash of chunks of newly arrived files with existing hash of chunks. If the hash of two chunks are matched, then public server will assume that the same file is already available at server and new chunk not get uploaded on the public cloud server only link to that data is to be provide to the file user. If new file is to be added to the cloud server and it get match the hash function of the old file then it only remove the new file and just provide hard link to the old file resides on the cloud server.

At the time of downloading, user only download the remaining chunks of file other than the chunks provided with links. Public server use the merge operation to provide all chunks in single file.

B. Mathematical Model

1. User Registration

$$RU = \{ru_1, ru_2, \ldots, ru_n\}$$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

Where, RU is the set of registered users. To become a part of system, user has to successfully register to system, by entering his valid details, such as user name, email ID, Contact no etc.

2. User Login

$$L = \{UN, PWD\}$$

Where,
L = Login, UN = User name, PWD = Password
To access the facilities of system such as store and retrieve the files on cloud server, user has to log into system.

3. RSA Key Generation:
After successful login, for each user, a key pair is allocated which is used for file encryption, decryption and token generation.
RSA algorithm is used for key generation.

$$KeyGen = \{PR_k, PUB_k\}$$

Where,
PUB_k = Public key = (n, e)
PR_k = Private Key = d = e mod (**p**-1)*(**q**-1)
Where,

p, q = Two large primes
n = product of p and q = p*q
e = special number greater than 1 and less than (p-1)*(q-1). $\{1 < e < [(p-1)*(q-1)]\}$

Encryption: $C = M^e \bmod n$
Where,
C= Ciphertext,
M = original Message
Decryption: $M = C^d \bmod n$

4. Token generation
The Key generated by RSA algorithm is encrypted by using AES algorithm and that encrypted key is Consider as a Token. Token is requested by user to public server to check the duplication of data stored at public server.

$$Token = AES\_Enc(d)$$

5. Chunking
Before storing the files on public server, Files are divided into chunks such as,

$$F = \{FC_1, FC_2, \dots, FC_n\}$$

F is the file and
FC is the chunk of that file.
The file is divided into chunk and each chunk = 10mb.
FC = 10 mb.

File divide into chunks:

```
Intchunk_size = 1024 * 1024 * 10;
File f = new file(Filename);
Long file_size = f.length();
f(size) = n mb;

Chunk c = f (size) mod (10 kb);
New_file = f (size) – Chunk c (size).
```

6. Hashing
   Hash of chunks are generated by using MD5 algorithm.
   F(X,Y,Z) = (X & Y) | (~(X) & Z)
   G(X,Y,Z) = (X & Z) | (Y & ~(Z))
   H(X,Y,Z) = X ^ Y ^ Z
   I(X,Y,Z) = Y ^ (X | ~(Z))
   Where &, |, ^, and ~ are the bit-wise AND, OR, XOR, and NOT operators

7. Encryption
   Each chunk is encrypted before storing on private server by using AES algorithm to provide the security over data.

   $$Enc(FC) = C_{FC}$$

   Where,
   C is the cipher text of chunk of file F, (FC).

   Suppose F is the file divided into n number of chunks$(FC_1, FC_2, \dots, FC_n)$, then the encrypted chunks $(C_{FC1}, C_{FC2}, \dots, C_{FCn})$ are stored on public server to provide the security over data.

8. Deduplication Checking

```
H(New chunk) = h
H(Old n chunks) = $h_n$
Compare h and $h_n$
If H(New chunk) == H(Old n chunks[])
        Chunk is duplicate and refuse it to store on public server
Else
        Chunk is not duplicate and allowed to store on public server
```

9. File download and decryption
   When user wants to download the file from public cloud server, server return the file by merging of all encrypted chunks of file. The chunks are identified by its ID.

   Whole Encrypted File = Merge$(C_{FC1}, C_{FC2}, \dots, C_{FCn})$

   Chunk merging to get original file:

   Num = get part counter of original file
   f.write(part counter)

   Then by using decryption algorithm user can get original file:
   $Dec(\text{Merge}(C_{FC1}, C_{FC2}, \dots, C_{FCn})) =$ Dec (C_{FC}) = F = original file.

C. Algorithms

Input: File Upload request from user with token.
Output: Data duplication checking
Process:
1. Get (Private key, public key) pair from private server.
2. Token request to private server
3. Token Generation and distribute to authorized users

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

```
4.  Get token and send file to public server
5.  At public server
6.  Get file and user authorization with token
7.  Divide file into 10 mb of chunks
8.  Generate hash of all chunks
9.  Compare h and $h_n$
10. If H(New chunk) == H(Old n chunks[])
11. Chunk is duplicate and not get uploaded
12. Else chunk is not duplicate get uploaded
```

IV. Results and Discussion

A. Experimental Setup

We implement our enhanced and secure chunk level deduplication system by considering three separate machines with JAVA programs such as, a program for client machine, for private server and for public server. With client machine program, user can requesting for file upload or download request to cloud data servers. With private server machine program, we can manage key and token generation, distribution and management as per private cloud. And on public server machine as a public cloud, we perform actual chunk generation and deduplication checking.

This system makes use of HTTP protocol for communication among three entities as discussed above. We perform the experiments on three Intel Core machine and there is a need of 4GB RAM with installed 32 bit Windows operating system. All machines are connected in Ethernet network.

B. Dataset Description

To evaluate the proposed chunk level deduplication system, we prepare two separate data sets. For each set we take 50 files where each of file does not exceed 100 MB size. Initially we upload first set of 50 files on public server. After this, we make some little modification in some files from set of 50 files and keep it as a second set of 50 files. For actual performance evaluation, we now upload this second set of files and check the chunk level deduplication performance of system.

C. Results Disscussion

Figure 2 depicts the comparison of 3 systems based on memory utilized after file uploading. For this we conduct three experiments. In first system, we just upload the files on public server, in second we upload the file on public cloud server with file level deduplication checking and in third as our proposed system, we upload files on private server with chunk level deduplication checking.

Among these three system, file upload with chunk level deduplication checking consumes less memory. Because in first syste, files are just uploaded on server without any deduplication checking, because of this duplicate files are also uploaded on server, which results in high memory wastage with duplicate file storage.

In second, only file level deduplication is performed. That is when new files arrives to server, it only check the name of files. That is is the files with same name is available on server then and only then it rejects for uploading. Sometimes, two files may have same data but has different names. In this situation actual data deduplication is not achieve.

To overcome this limitation, we proposed chunk level dedupliaction. In this, content wise deduplication is performed. Suppose one file is already stored on server. Next time user modify some data of this file and wants to store it again on server. Then instead of storing whole files, server will divide that files into chunks and perform chunk level deduplication. If the data of chunks of new files is not matched with already stored chunk data, then server allows that chunk for uploading. In this way, we achieve the high level of memory wastage. And because of this, among three system our proposed system is better in terms of memory savings.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

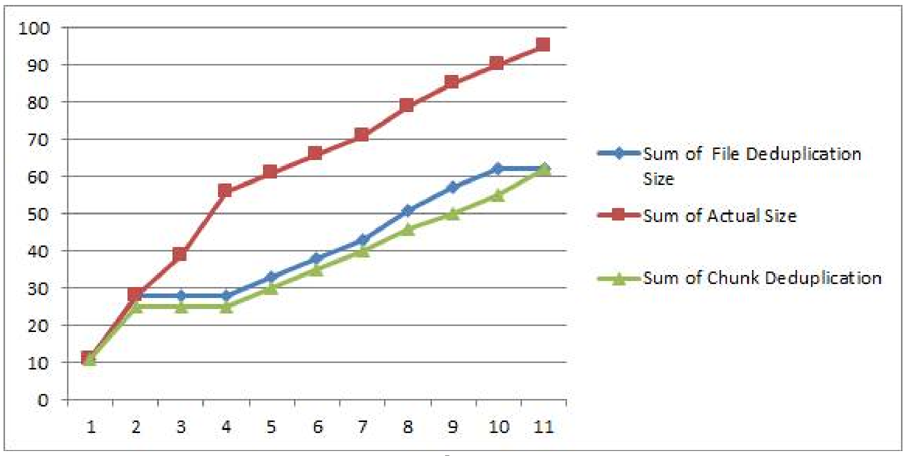

Fig. 2 Memory Utilization Comparison

Figure 3 depicts the time required by three systems such as Actual file upload, Upload with file level deduplication and Upload with chunk level deduplication. From figure it is clear that the minimum time is taken by proposed system for whole procedure. In first system, all files are uploading so it took more time for file upload. In second system, if the file name is not same then upload file. So user can just modify the name so that it also took maximum amount of time. But in proposed system, only unique chunks are going to upload on server therefore it took less time among three systems.

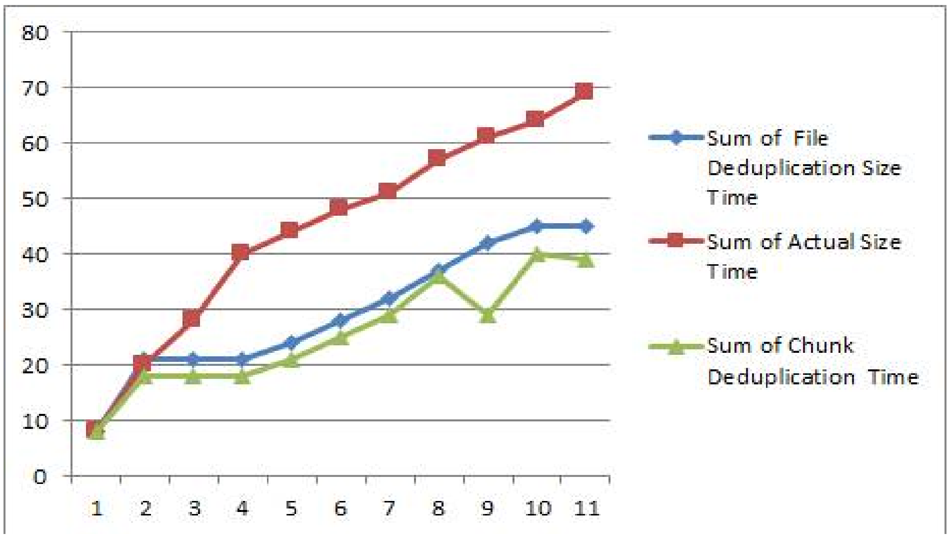

Figure. 3 Time Comparison

V. Conclusion and Future Work

This paper shows that the proposed method for data deduplication is authorized and securely duplication of the file is done. In this we have also proposed new duplication check method which generate the token for the private file. As a proof of ownership of the data user need to submit the key along with the file upload request. We have solved more critical part of the data deduplication checking at cloud data storage with chunk level deduplication and hash of chunks. From experimental evaluation, we prove that the proposed system with encryption and chunk level deduplication checking, achieves high security over data and also efficient in terms of memory and time of processing file upload requests.

REFERENCES

1. M. Bellare, S. Keelveedhi, and T. Ristenpart. Dupless: Serveraided encryption for deduplicated storage. In USENIX Security Symposium, 2013.
2. P. Anderson and L. Zhang. Fast and secure laptop backups with encrypted de-duplication. In Proc. of USENIX LISA, 2010.
3. J. Li, X. Chen, M. Li, J. Li, P. Lee, andW. Lou. Secure deduplication with efficient and reliable convergent key management. In IEEE Transactions on Parallel and Distributed Systems, 2013.
4. S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of ownership in remote storage systems. In Y. Chen, G. Danezis, and V. Shmatikov, editors, ACM Conference on Computer and Communications Security, pages 491–500. ACM, 2011.
5. J. Li, X. Chen, M. Li, J. Li, P. Lee, andW. Lou. Secure deduplication with efficient and reliable convergent key management. In IEEE Transactions on Parallel and Distributed Systems, 2013.
6. C. Ng and P. Lee. Revdedup: A reverse deduplication storage system optimized for reads to latest backups. In Proc. of APSYS, Apr 2013.
7. C.-K Huang, L.-F Chien, and Y.-J Oyang, “Relevant Term Suggestion in Interactive Web Search Based on Contextual Information in Query Session Logs,” J. Am. Soc. for Information science and Technology, vol. 54, no. 7, pp. 638-649, 2003.
8. S. Bugiel, S. Nurnberger, A. Sadeghi, and T. Schneider. Twin clouds: An architecture for secure cloud computing. In Workshop on Cryptography and Security in Clouds (WCSC 2011), 2011.
9. W. K. Ng, Y. Wen, and H. Zhu. Private data deduplication protocols in cloud storage. In S. Ossowski and P. Lecca, editors, Proceedings of the 27th Annual ACM Symposium on Applied Computing, pages 441– 446. ACM, 2012.
10. R. D. Pietro and A. Sorniotti. Boosting efficiency and security in proof of ownership for deduplication. In H. Y. Youm and Y. Won, editors, ACM Symposium on Information, Computer and communications Security, pages 81–82. ACM.
11. S. Quinlan and S. Dorward. Venti: a new approach to archival storage. In Proc. USENIX FAST, Jan 2002.
12. A. Rahumed, H. C. H. Chen, Y. Tang, P. P. C. Lee, and J. C. S. Lui. A secure cloud backup system with assured deletion and version control. In 3rd International Workshop on Security in Cloud Computing, 2011.

BIOGRAPHY

SunitaVelapure received the BE in Information Technology from Aditya college of engineering which is under Dr. BAMU University, Aurangabad. Currently pursuing ME in Information Technology from SKN College of engineering, Pune, India.