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. 3, Issue 3, March 2015

Efficient Deduplication with Security Using Jenkins and Recovery Techniques

S.Hemalatha[1], U.Muthaiah[2]

P.G Scholar, Department of CSE, Sri Shanmugha College of Engineering and Technology, Pullipalayam, Salem (Dt), India[1]

Assistant Professor, Department of CSE, Sri Shanmugha College of Engineering and Technology, Pullipalayam, Salem (Dt), India[2]

ABSTRACT: Data deduplication is a method of tumbling stowage needs by eliminating redundant data. Only one unique occasion of the data is actually retained on storage broadcasting. Redundant data is replaced with a pointer to the unique data copy and it has been widely used in cloud packing to reduce the amount of storage space and save bandwidth. To protect the concealment of sensitive data while supporting deduplication, Jenkins hash function (JHF) techniques has been proposed to encrypt the data before outsourcing. To enhanced protect data security, it makes the first endeavour to formally address the problem of authorized data deduplication. The several deduplication techniques are implemented in hybrid Cloud architecture. In additionally implement the Log-based Recovery technique in hybrid cloud. Our proposed system is implemented for text file, pdf file, book,image and video, it also verify the confidentiality of private cloud. Simulation result shows that the proposed system authorized duplicate check scheme incurs minimal overhead when compared to normal operations and also it decreases the authorization parameters to above 60%.

KEYWORDS: Deduplication, authorized duplicate check, confidentiality, hybrid cloud,Merkle, Jenkins key.

I. INTRODUCTION

Cloud computing provides seemingly unlimited virtualized resources to users as services across the whole Internet, while whacking platform and implementation details. Today’s cloud service provider’s proposal both highly available storage and massively parallel computing resources at relatively low costs. As cloud computing becomes rampant, an increasing amount of data is being stored in the cloud and shared by users with itemized privileges, which define the access rights of the stored data. One precarious challenge of cloud storage[4] services is the management of the ever-increasing volume of data. To make data management accessible in cloud computing, deduplication has been well known technique and has attracted more and more devotion recently. Data deduplication[10] is a specialized data compression technique for eliminating duplicate copies of repeating data in storage, the technique is used to progress storage.

Utilization and can also be applied to network data transfers to reduce the number of bytes that must be sent. Instead of keeping several data copies with the same content, deduplication eradicates redundant data by keeping only one physical copy and referring other redundant data to that copy[2]. Deduplication can take place at whichever the file level or the block level. For file level deduplication, it eradicates duplicate copies of the alike file. Deduplication can also take place at the block level, which rejects duplicate blocks of data that occur in non-identical files.

Although data deduplication brings a lot of benefits, security and solitude anxieties arise as users’ sensitive data are susceptible to both inside and outside attacks. Thus, identical data copies of different users will lead to different cipher texts, making deduplication impossible[1]. Jenkins encryption has been proposed to enforce data confidentiality while making deduplication viable. It encrypts/ decrypts a data copy with a Jenkins key, which is obtained by computing the cryptographic hash value of the content of the data facsimile. After key generation and data encryption, users preserve the keys and send the cipher text to the cloud. Since the encryption operation is daunt monistic and is derived from the data satisfied, identical data copies will engender the same Jenkins key and hence the equal cipher text.

10.15680/ijircce.2015.0303082

To avert unauthorized access, a secure proof of ownership protocol is also needed to provide the proof that the user indeed owns the equivalent file when a spare is found. Before acquiescing his duplicate check request for specific file, the user needs to take this file and his own privileges as inputs [1].

The section II describes the related works about this project and section III and IV describes proposed system and its implementation and finally section V describes experimental result and its enhancement.

II. RELATED WORK

Recent years have witnessed the trend of leveraging cloud-based services for large scale gratified storage, processing, and distribution. Security and privacy are among top concerns for the public cloud environments. That is, every client computes as per data key to encrypt the data that he intends to store in the cloud. As such, the data access is fared by the data owner[3]. Second, by assimilating access privileges in metadata file, a ratified user can decipher an encrypted file only with his private key.

A. System Model

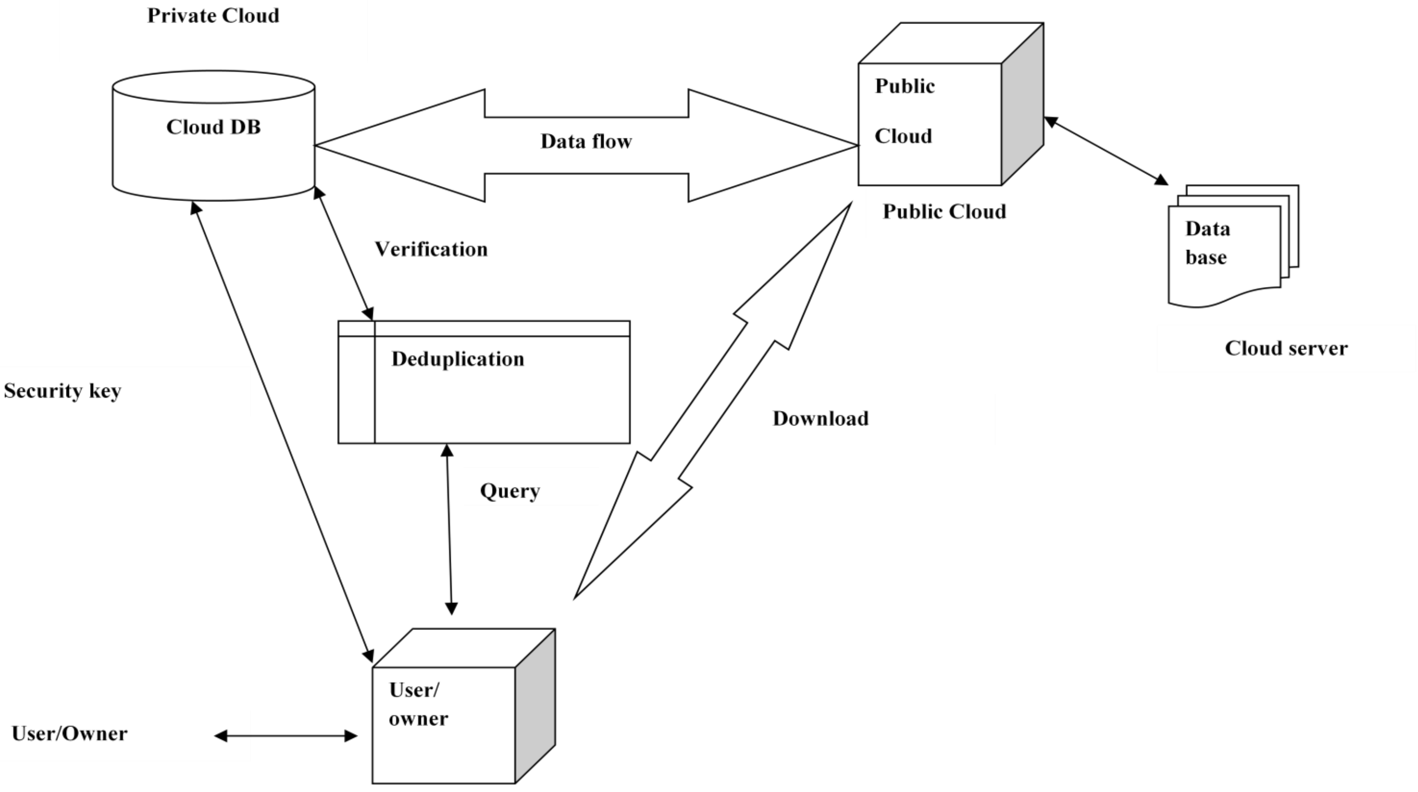

Fig 1.Deduplication checking process in hybrid cloud

This paper introduces a new cryptographic method for secure Proof of Ownership (PoW)[7], based on the joint use of Jenkins encryption and the Merkle-based Tree, for improving data security in cloud storage systems, providing dynamic sharing between users and ensuring efficient data deduplication. Our idea consists in using the Merkle-based Tree over encrypted data, in order to originate a distinctive identifier of subcontracted data. On one hand, this identifier serves to check the availability of the same data in remote cloud servers. On the other hand, it is used to ensure efficient access control in dynamic sharing scenarios. From the perspective of cloud storage security, there have been two notable notions:

B. *Proof of Data Possession (PDP)*

It allows a cloud client to verify the integrity of its data subcontracted to the cloud in a very efficient way. This is plausible because it could be very resource-consuming to load a large data file from secondary storage to memory.

C. *Proof of Retrievability (POR)*

This notion was introduced by Juels and Kaliski. This explains the term “deduplication”. This issue was first introduced to the research community. Because straightforward deduplication is vulnerable to attacks Halevi proposed the notion called Proof of Ownership (POW) as well as concrete constructions[7].

III. PROPOSED ALGORITHM

These represent the enhancement of proposed system in security. Jenkins has function techniques has been proposed to enforce data confidentiality while making deduplication feasible. . It encrypts/decrypts a data copy with a Jenkins key, which is obtained by computing the cryptographic hash value of the content of the data copy. After key cohort and data encryption, users preserve the keys and send the cipher text to the cloud. Jenkins key and hence the same cipher text. To prevent unauthorized contact, a secure proof of ownership protocol is also needed to provide the proof that the user indeed owns the same file when a duplicate is found. After the proof, ensuing users with the same file will be provided a pointer from the server without needing to upload the alike file. A user can download the encrypted file with the pointer from the attendant, which can only be decrypted by the corresponding data owners with their Jenkins keys[8].

A. *MODULES*

a. *Data Users*

A user is an entity that wants to outsource data storage to the public cloud and access the data later. For security use a cryptographic procedure called encryption using Jenkins hash function algorithm.

b. *Private Cloud*

The private keys for the privileges are managed by the private cloud, which answers the file perfunctory requests from the users. The edge accessible by the private cloud allows user to submit files and queries to be securely stored and computed respectively. Each file uploaded to the cloud is also bounded by a set of privileges to specify which kind of users is allowed to perform the duplicate check and access the files based on their set of privilege key provide to them [5].

c. *Jenkins Hash Function*

JHF technique**s** used to enforce data confidentiality. Jenkins encryption allows the cloud to perform deduplication on the cipher texts and the proof of ownership prevents the unauthorized user to access the file.

Algorithm

1. Take any one of the following user information.
   Username, password, mobile number, uploaded document name, document size
2. Convert bit to byte conversion
3. Byte conversion answer placed in 64*64 cells
4. Insertion order is not unique.
5. Re order the element and placed in outside the cell.
6. Finally add the mobile number or document size to reorder element

Hash Table Lookup

A hash table (hash map) is a data structure used to implement an associative array. A hash table uses a hash function to compute an index into an array of buckets or slots. In a well-dimensioned hash table, the average cost for each lookup is independent of the number of elements stored in the table. Many hash table designs also allow arbitrary insertions and deletions of key-value pairs, at constant average cost per operation.

Merkle Hash Tree

A hash tree or Merkle tree is a tree in which every non-leaf node is labelled with the hash of the labels of its children nodes. Hash trees can be used to verify any kind of data stored. Currently the main use of hash trees is to make sure that data blocks received from other peers in a peer-to-peer network are received undamaged and unaltered, and even to check that the other peers do not lie and send fake blocks.

d. Public Cloud

This is an entity that provides a data storage service in public cloud. The public cloud provides the data outsourcing service and stores data on behalf of the users. To reduce the storage cost, the public cloud eliminates the storage of redundant data via deduplication and keeps only unique data [6].

e. Log-Based Recovery

Log is a sequence of records, which maintains the records of actions performed by a transaction. It is important that the logs are written prior to actual modification and stored on a stable storage media, which is fail safe[9].

Log Based Recovery works as Follows

• When a transaction enters the system and starts execution, it writes a log about it
<Tn, Start>
• When the transaction modifies an item X, it write logs as follows:
<Tn, X, V1, V2>
It reads Tn has changed the value of X, from V1 to V2.
• When transaction finishes, it logs:
<Tn, commit>

Database can be modified using two approaches:

1. Deferred database modification: All logs are written on to the stable storage and database is updated when transaction commits.
2. Immediate database modification: Each log follows an actual database modification. That is, database is modified immediately after every operation.

IV. FILE UPLOAD PROCESS

File Tag (File)

It computes hash of the File as File Tag.

TokenReq (Tag, UserID)

It entreaties the private Server for File Token generation with the File Tag and User ID.

DupCheckReq (Token)

It appeals the Storage Server for Duplicate Check of the File by sending the file token established from private server.

ShareTokenReq (Tag, {Priv}):

It requests the Private Server to engender the Share File Token with the File Tag and Target Partaking Privilege Set.

FileEncrypt (File):

It scrambles the File with Jenkins Encryption.

FileUploadReq (FileID, File, Token):

It uploads the File Data to the Storage Server if the file is Inimitable and updates the file Token stored.

Private Server Program:

TokenGen (Tag, UserID):

It loads the concomitant privilege keys of the user and generate the token.

ShareTokenGen (Tag, Priv):

It engenders the share token with the corresponding privilege keys of the sharing privilege.

Storage Server program:

DupCheck (Token):

It explorations the File to Token Map for Duplicate.

FileStore (FileID, File, Token):

It provisions the File on Disk and updates the Mapping.

V. SIMULATION RESULTS

The time spent on tagging, encryption, upload increases linearly with the file size, since these operations involve the actual file data and incur file I/O with the whole file**.** In contrast, other steps such as token generation and duplicate check only use the file metadata for computation and therefore the time spent remains constant. In fig 4.1 with the file size increasing from 10MB to 400MB, the overhead of the proposed authorization steps decreases from 0.483%.to 0.283%

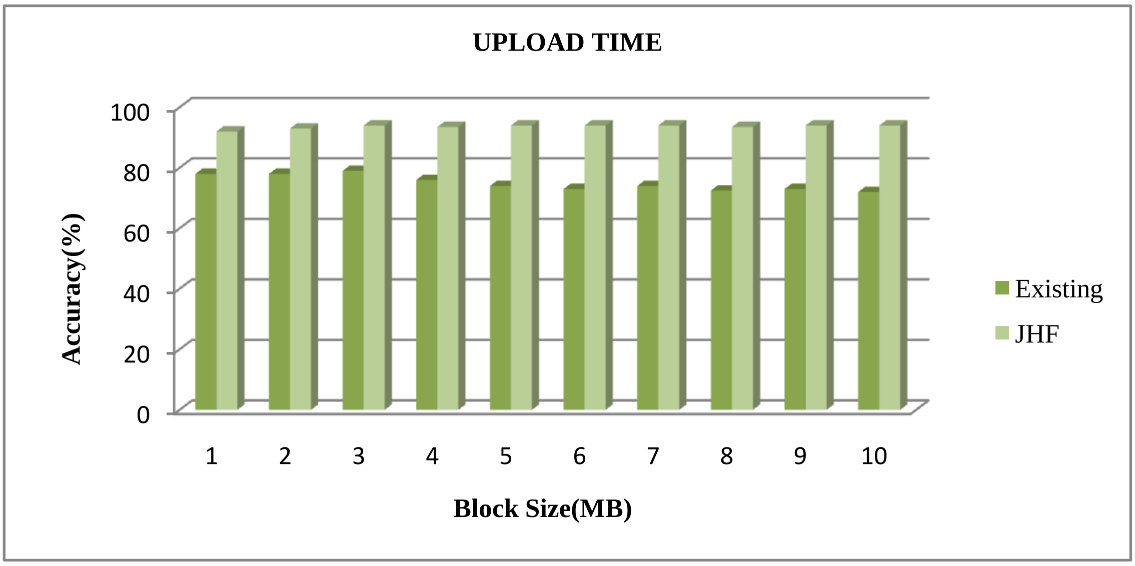

Fig 2.Upload time of file

To evaluate the effect of the deduplication ratio, prepare two unique data sets, each of which consists of 50 100MB files. First upload the first set as an initial upload. For the second upload, pick a portion of 50 files, according to the given that deduplication ratio, from the initial set as facsimile files and remaining files from the second set as inimitable files. The average time of uploading the second set is presented in Figure 4.2 As uploading and encryption would be skipped in case of duplicate files, the time spent on both of them diminutions with increasing the deduplication ratio. The time spent on duplicate check also diminutions as the searching would be ended when facsimile is found. Total time spent on uploading the file with deduplication ratio at 100% is only18.5% with unique files.

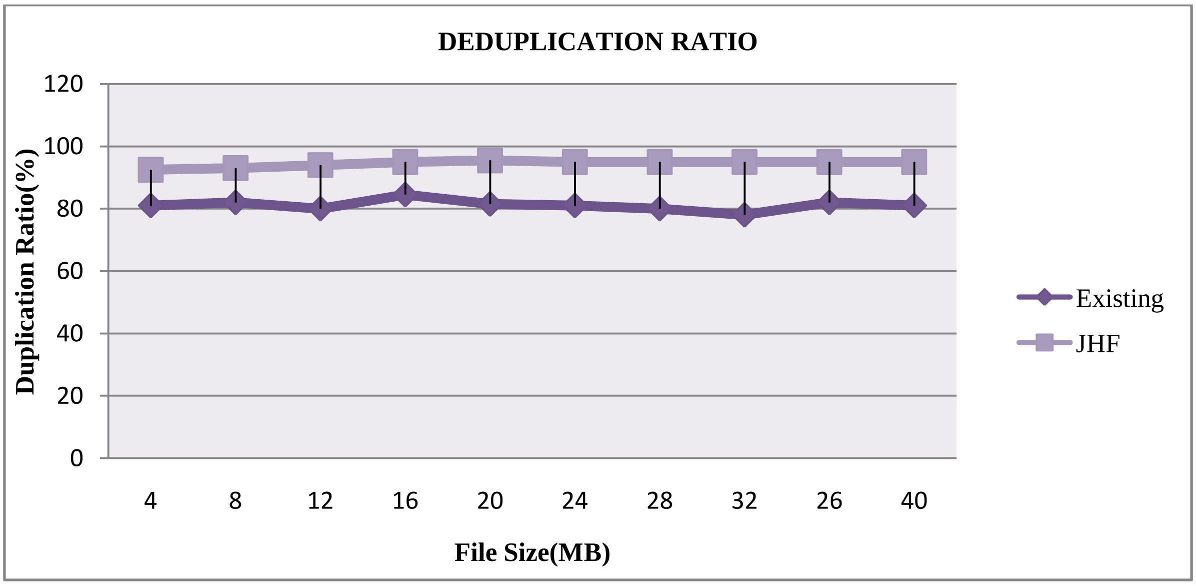

Fig 3.Deduplication Ratio

VI. CONCLUSION AND FUTURE WORK

As a proof of concept, implemented a prototype of our proposed authorized duplicate check scheme and conduct test bed experiments on our prototype. Showed that our authorized duplicate check scheme incurs minimal overhead compared to Jenkins Hash function and network transfer. To further enhance, need to check the deduplication for films uploading, books with large number of pages in efficient time. Also to search for effective algorithm that works faster than JHF.

REFERENCES

1. W. K. Ng, Y. Wen, and H. Zhu. Private data deduplication protocols in cloud storage. InS. Ossowski and P. Lecca, editors, Proceedings of the 27th Annual ACM Symposium 2013.
2. J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl. A secure data deduplication scheme for cloud storage. In Technical Report, 2013.
3. J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou. Secure deduplication with efficient and reliable convergent key management. In IEEE Transactions on Parallel and Distributed Systems, 2013.
4. J. Yuan and S. Yu ,Secure and Efficient Proof of Storage with Deduplication. Secure and efficient proof of storage with deduplication. IACR Cryptology ePrint Archive, 2013:149, 2013.
5. Chang Liu Nesrine Lauren France, fnesrine kaaniche, A Secure Client Side Deduplication Scheme in Cloud Storage Environments maryline.laurentg@telecom-sudparis.eu 2012.
6. Jiawei Yuan, Shucheng Yu ,Secure and Constant Cost Public Cloud Storage Auditing with Deduplication , Department of Computer Science University of Arkansas at Little Rock, USA Email: sxyu1@ualr.edu 2012
7. 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.
8. C. Botelho Gerais Belo Horizonte,br,Yoshiharu Kohayakawa , An Approach for Minimal Perfect Hash Functions for Very Large Databases Fabiano Dept. of Computer Science Univ. of Sa˜o Paulo, Brazil yoshi@ime.usp.br.2011
9. P. Anderson and Zhang ,Fast and Secure Laptop Backups with Encrypted deduplication. In Proc. of USENIX LISA, 2010.
10. M. W. Storer, K. Greenan, D. D. E. Long, and E. L. Miller. Secure data deduplication.In Proc. of StorageSS, 2008.

BIOGRAPHY

1. Chang Liu Department of Computer Science. His research interest in Programming Language, Security, Knowledge Representation, Semantic Web, Database, Distributed System. Working experience in Microsoft Research, Redmond, Intern, University of Maryland, Research Assistant, IBM China Research Lab, Intern Awards won: Best Paper Award, ASPLOS 2015, SoCC ’14 Student Scholarship, Programming Language Mentoring Workshop Schoarship Award, 2015, 2013 The NSA Best Scientific Cyber security Paper Award, IEEE Symposium on Security and Privacy 2014 Student Travel Grants.

10.15680/ijircce.2015.0303082