International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2014): 5.611

# Secure Deduplication for Cloud Storage

**Minal Chandrashekhar Pore.[1], Nilav Mukhopadhyay[2]**

[1]Department of Computer Engineering, ME(CN II), Dr. D. Y Pati School Of Engineering & Technology, Lohegaon, Pune, India

[2]Assistant Professor, Department of Computer Engineering, Dr. D. Y. Patil School of Engineering & Technology, Lohegaon, Pune

**Abstract:** ***Cloud computing provides unlimited storage space, availability, accessibility from anywhere, anytime to individuals. Due to continuously increase in number of users and their data. Data deduplication becomes important technique for cloud storage providers. Data duplication removes the redundancy and replication, but confidentiality and secrecy of data becomes important issue. Data deduplication stores only one redundant copy of data and provides link to that copy. Two popular available data deduplication strategies are file wise deduplication and block level deduplication. Secure data deduplication allows cloud storage and service providers to employ data with confidentiality.***

**Keywords:** Cloud storage, Data deduplication, Data security, Block level duplication, File level duplication.

## 1. Introduction

Cloud environment offers scalable and elastic storage capabilities to users. Data deduplication is widely used technique to reduce the storage spaces and saving the bandwidth. Data deduplication stores only one copy of redundant data, and provides link to that copy instead of storing again. It acts as a key component in backup process. Deduplication saves both the disk space and network bandwidth. Data deduplication strategies classified as 1) File level deduplication- Here only a single copy of each file is stored on the server. 2) Block level deduplication-

Each file is divided into blocks and only single copy of each block is stored [2]. Block size can be either fixed or variable. From the architecture perspective deduplication can be target based or source based.1) Source based deduplication- Before uploading data user first sends identifier to the server for redundancy checking. 2) Target based deduplication- Users are unaware of deduplication, they just upload the files to the storage server. In this paper, we are focusing on deduplication with security.

But, encryption and dedupication are conflicting technologies. Deduplication detects identical segments and storing them only once, while encryption makes two identical data segments indistinguishable after encryption.

Convergent encryption technique has been proposed to meet these conflicting requirements. Convergent encryption technique uses content hash keying [3]. It is used in cloud computing to remove duplicate files from storage without provider having access to encryption keys. Convergent technique provides confidentiality and deduplication at same time but it is susceptible to dictionary attacks [6].

Authenticated and anonymous approach is followed for secure deduplcation. These models differ in security properties and can be applied to single server storage as well as distributed storage. In single side storage, both data and metadata are stored on single server where as in distributed environment separate servers are maintained for storing metadata and data. To store data object based storage devices are used.

## 2. Related Work

In deduplication process first file is divided into smaller units and this mechanism is known as chunking. Small units are called as chunks. Whole file, fixed size and variable size chunks are three basic approaches for deduplication. In whole file strategy file's hash value is computed and used as identifier. Two files having same hash value considered as identical and only one copy is stored. Such form is known as content addressable storage and used in EMC centera systems [1]. In second approach files are divided into fixed size blocks before deduplication. Third one is most suitable approach in which files are break into variable size chunks by manipulating hash value for particular file.

To provide data secrecy along with deduplication many storage providers use convergent encryption. In CE hashed value of file is used as encryption key. Convergent encryption is considered as five tuple polynomial algorithm. From original data copies convergent key is generated by user. Then tag is generated for checking duplicate copies [13]. Before encryption, some storage providers use content defined chunking to divide the file.

Such type of algorithm depends on the content of file. CDC uses basic sliding window approach. BSC algorithm scans the content and fingerprints it.

To prevent the data leakage Proof of ownership approach is introduced. In this cloud server is able to check user data ownership based on computed short value. To avoid unauthorized access POW protocol [13] is implemented. If duplicate file is found, it provides a proof for user to confirm user owns the same file. After proof, one pointer is provided to user using which user can download encrypted file from server. Further file is encrypted by data owners using convergent keys. Hence, convergent Encryption provides deduplication for cloud on cipher text and POW allows only authorized user to access the encrypted file.


Licensed Under Creative Commons Attribution CC BY

## 3. Threat Model

To evaluate the security of storage system, threat model should be cleared out. For building the security model consistent notations should be established. In this paper we are using the concepts of encryption and hashing. Encryption is cryptographic function which transforms the plaintext into cipher text. Encryption function can be denoted as e (K, P) =C., Where P is given plaintext and k denotes the key using which cipher text C is generated. For every encryption function there is corresponding decryption function which restores the plaintext from cipher text using key. It can be denoted as d (K, C) =P.

Hash function is encryption function which maps the arbitrary data to fixed size value. It can be denoted as h(P) = H where H is hashed value. The process of creating hash code of plaintext using any key $k_i$can be considered as keyed Hash Message Authenticate Code (HMAC), which can be given as $HMAC_i$ (P).

## 4. Primary Players

Storage model has three primary players' client, metadata store, and chunk store. We can map this model to single storage as well as distributed storage system. In single server storage metadata and chunk store both are stored on the same system. While in distributed server storage metadata store and chunk store are disconnected [7] [8].

Client is starting point for both ingestion and extraction using which users interact with system. Figure shows that client acts as a Central contact. It does not need persistent storage. Metadata store maintains the information which is required to users to rebuild files from the chunks for e.g. keys and maps. This can be managed with persistent, non-verified key valued architecture where user submits the key and value pair to server. Chunk store persistently stores the data chunks and manages requests made for chunks based on chunk ID. Chunk store verifies the correctness of value with key. This should be done to avoid target collision attacks. Along with these three players system considers internal and external adversaries. External adversaries [10] are outside attackers and don't even have access to user accounts. Internal attackers are harmful because of their existence chunk store and metadata store cannot be considered trustworthy.

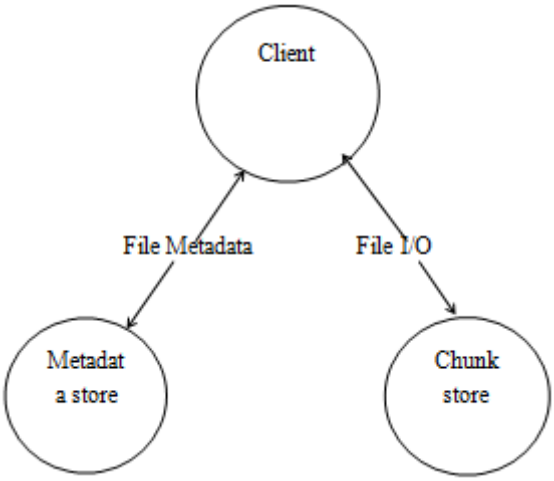


**Figure 1:** Three Primary Players in Storage Model

## 5. System Design

The goal of model is to provide data protection and security from both attackers without considering access levels [6]. Both models given in paper offer data secrecy against external and internal adversaries. Anonymity provides concealment for user identity. The process of removing user's access is termed as revocation. After revocation user is unable to see the changes but has continuous access to view what they have entitled previously.

### 5.1 Overview of Secure Deduplication

Client starts the ingestion process by breaking file into set of chunks. Client can use content based chunking to break the file. This approach can match the shared content whether content exists or not multiple times with given offset. The chunks are selected according to threshold value and width of sliding window. If $F_{k,k+w-1}>A$ , then k can be selected as chunk boundary. A is threshold value. At each k position fingerprint $F_{k,k+w-1}>A$ is calculated. Set of variable sized chunks is obtained as a result Set. File chunking and encryption both are performed on client side, due to which processing efforts at server are reduced. Because of encryption data is not sent in clear format which minimizes the external attacks. Client uses convergent encryption to encrypt the chunks. Regardless of whom does the encryption due to identical key given plaintext produce similar cipher text? K= hash (chunk) strategy offers number of advantages. But here key sharing problem occurs due to sharing of random key across several users. One important benefit of scheme is that even root administrator also does not have access to chunk's plaintext without knowing key value. Primary disadvantage of this approach is information leakage [6]. But to achieve space efficiency such behavior is compromised in deduplication environment. Encrypted chunk's hash value is used to identify the chunk and technique is known as content based naming.

Chunk_id=hash (e (hash (chunk),chunk))

We can also use hash of encryption key as chunk identifier, due to which performance is improved.

### 5.2 Authenticated model

Authenticated model considers number of terms regarding encryption keys and the key management techniques available to users. First consider that each user has private key and pair of asymmetric key. There is certificate authority which is responsible for trusted distribution of public keys. Users generate the strong encryption keys. After identifying chunks client encrypts them using convergent encryption and starts the ingestion process. Chunk map is responsible for maintaining all the information required to rebuild the files, including locations for chunks and keys. This map is stored in metadata store and accessed via inode number of particular file. For better security it is encrypted by map key. Only authorized users can decrypt the map with the help of public key [9]. Map key is appended to map entry as more users are granted access to file. Encrypted chunks are stored to chunk store after ingestion process. Chunk store generates the ID for extraction.


Licensed Under Creative Commons Attribution CC BY

When client authenticated by the metadata store and submit Open () request extraction process starts. Encrypted chunk map and list of map keys can be located via file's inode number. Only the chunk map and key which corresponds to particular user is returned by metadata store instead of returning chunk map and whole list of keys. Finally client decrypts map key and chunk maps. List of encrypted map keys is important for revocation.

New chunk can be generated after revocation and chunk map is encrypted using new key.

### 5.3 Anonymous Model

Anonymous model attempts to hide details of authors and readers. The model considers that encrypted data is secure against adversary. The limitation of anonymous data store is that both legitimate and malicious users are anonymous. To avoid attacks both metadata and chunk store kept immutable. Due to this changes made in file are reflected in versioned copy of chunk map, so malicious changes can be isolated. This model considers the symmetric encryption i.e. key is private to user. The particular key is used in HMAC procedures so that only owner confirms the hash [7].

In anonymous model, ingestion begins with identification and encryption of chunks. Map reference allows multiple users to see changes made to file by authorized user. Access to particular file is granted outside by sharing map key. Users create and store map reference in metadata store. If changes are made to file, new map entry is written to metadata store as linked list. Each user's map reference is used to locate the root linked list with the help of HMAC which is keyed with map key. After committing write by client new node is appended to list. File sharing is done using authorized key.

File extraction process requires four stages for completion. First, to obtain particular file's inode client access metadata store and issue open () request. Using map reference client obtains the map key. During the third phase linked list is traversed by client until they reach to intend version of map entry or till end of list indicating request fails. In last phase client determines which chunk to request from store based on map entry.

## 6. Adversaries

To evaluate the security of models we examine the external attacks as well as malicious insiders who have access to system data [10].Also we have to consider key security for security implications. For a system to be considered as secure it should not leak any information outside. In case of external adversaries passive attackers aim to obtain information in transit, but active attackers modify the information content. System must be protected from internal adversaries. Internal attackers are harmful as they have information about system. Malicious insider with full access can modify or delete the sensitive data.

## 7. Conclusion

In this paper two models are considered for secure deduplication. To protect confidentiality of data in case of deduplication environment is important. Security is provided using convergent encryption technique to encrypt the data. For both models map is created using which file can be reconstructed. In authenticated model sharing of map key is managed through asymmetric key pairs. In anonymous model, storage is immutable and map references are created for file sharing. We consider security implications regarding system. For any system to be considered as secure adversaries are considered. To reduce the storage we must have considered deduplication along with security.

## References

[1] S. Annapureddy, M. J. Freedman, and D. Mazières. Shark: Scaling file servers via cooperative caching. In Proceedingsof the 2nd Symposium on Networked Systems Design andImplementation (NSDI), pages 129–142, 2005.

[2] Harnik D.Benny Pinkas Alexandra Shulman-Peleg. Cloud Computing: Side Channels in cloud services. IEEE Security and Privacy, 8(6):40–47, 2010.

[3] Pasquale Puzio Refik Molva Melek Sergio Onen L. ClouDedup: Secure Deduplication with Encrypted Data for cloud storage. In Proceeding in 2013 IEEE International Conference on Cloud Computing Technology and Science

[4] I. Clarke, O. Sandberg, B.Wiley, and Hong T.W. Freenet: A distributed anonymous information storage and retrieval system. Lecture Notes in Computer Science, 2009:46–66, 2001.

[5] D. Bhagwat, K. Pollack, D. D. E. Long, E. L. Miller, J.-F. Pâris, and T. Schwarz, S. J. Providing high reliability in a minimum redundancy archival storage system. In Proceedings of the 14th International Symposium onModeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '06), Monterey, CA,Sept. 2006.

[6] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In Proceedings of the 22nd International Conference on Distributed Computing Systems(ICDCS '02), pages 617–624, Vienna, Austria, July 2002.

[7] Kallahalla M., E. Riedel, R. Swaminathan, Q. Wang, and K. Fu. Plutus: scalable secure file sharing on untrusted storage. In Proceedings of the Second USENIX Conferenceon File and Storage Technologies (FAST), pages 29–42, San Francisco, CA, Mar. 2003. USENIX.

[8] J. Yuan and S. Yu, "Secure and constant cost public cloud storage auditing with deduplication," IACR Cryptology ePrint Archive, 2013:149, 2013

[9] E. L. Miller, D. D. E. Long, W. E. Freeman, and B. C. Reed. Strong security for network-attached storage. In Proceedingsof the 2002 Conference on File and Storage Technologies (FAST), pages 1–13, Monterey, CA, Jan. 2002.

[10] D. Goldschlag, M. Reed, and P. Syverson. Onion routing. Communications of the ACM, 1999.

[11] F. Douglis and A. Iyengar. Application-specific delta-encoding via resemblance detection. In Proceedings ofthe 2003 USENIX Annual Technical Conference, pages 113–126. USENIX, June 2003.


Licensed Under Creative Commons Attribution CC BY

[12] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage," Tech. Rep. IBM Research, Zurich, ZUR 1308-022, 2013.
[13] J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," in Proc. IEEE Trans. ParallelDistrib.Syst.,http://doi.ieeecomputersociety.org/ 10.1109/TPDS.2013.284, 2013.
[14] M. Bellare, S. Keelveedhi,and T. Ristenpart, "Dupless: Serveraided encryption for deduplicated storage," in Proc. 22nd USENIX Conf. Sec. Symp., 2013, pp. 179–194.
[15] L. L. You, K. T. Pollack, and D. D. E. Long. Deep Store: An archival storage system architecture. In Proceedings of the21st International Conference on Data Engineering (ICDE '05), Tokyo, Japan, Apr. 2005.

www.ijsr.net
IJSR
ISSN (Online): 2319-7064