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

Secure Manipulation of Deduplicated Data for Cloud Storage

Renuka Devi S[1], Eldo P Elias[2]

[1]PG Student, Department of Computer Science and Engineering, Mar Athanasius College of EngineeringKothamangalam, Kerala

[2]Assistant Professor, Department of Computer Science and Engineering, Mar Athanasius College of Engineering Kothamangalam, Kerala

Abstract: ***Cloud storage enables to configure and use online storage devices as storage targets. The increasing demand of cloud storage requirement has led to the process of deduplication. The term data deduplication refers to the techniques that store single copy of redundant data, and provide links to that copy instead of storing other actual copies of this data. Deduplication enables to reduce cloud storage space usage and uses network bandwidth efficiently. Convergent encryption technique is used to implement deduplication in the cloud storage, but it introduces new security threats. There comes the need of secure deduplication. Security of the proposed system relies on a proxy server which applies additional encryption on the uploaded data; this protects the data against convergent encryption attacks. System also includes a deduplication manager which deals with block level deduplication. Any cloud storage provider can be easily integrated in the new architecture. The system is implemented with the help of JAVA programming language, SQL database and Amazon s3 cloud service and the results are analyzed.***

Keywords: Deduplication, Cloud Storage, Convergent Encryption.

1. Introduction

Data deduplication is an advanced technology in a data backup system. It provides an efficient method to transmit and store data by identifying and eliminating duplicate blocks of data during backups [1].In de-duplication, only single copy of the data is actually stored on the server and duplicate data is replaced with a pointer to the unique data copy. Deduplication helps to utilize the available storage space efficiently. It also helps to reduce storage cost and bandwidth cost. Deduplication has several applications in the areas of cloud data center, web data center and email data center. This paper explains the deduplication within cloud storage area.

Deduplication process protects the cloud server from storing redundant data. If two users want to upload the same file, only a single file will be uploaded on the cloud server and the users will be provided with a link that will fetch the whole file for them whenever they want to retrieve it. Suppose user1 on cloud stores a file A. He will request to upload the file and the file will be successfully uploaded now when a user, user2 uploads the same file, the cloud will deduplicate the file by providing user2, the link of file A which is already present on the cloud. Thus „n" users are allowed to access same file with a single copy stored on cloud. Since in the cloud storage users pay according to the amount of capacity being used, reducing file sizes before transferring to the cloud helps to achieve large savings.

Deduplication technique works by partitioning data into chunks of non-overlapping data blocks. Then calculates a hash value for each chunk and stores that value in a hash table. Cryptographic hash function (e.g. SHA 1) can be used for generating hash value. To determine whether a chunk is already stored on the system or not, the hash value of the incoming data item is first looked up in the table and if there is a match, the system only stores a reference to the existing data. Otherwise the incoming chunk is considered unique and is stored on the system and its hash value is inserted into the hash table. File level de-duplication searches for identical file whereas block level de-duplication eliminates identical blocks within different files. When the deduplication occurs close to where the data is created, it is referred to as source deduplication (before transferring data to the cloud and is usually performed at client side). When it occurs near where the data is stored, it is called target deduplication (performed by server on the cloud data).

While there are several advantages of the cloud, data security remains a top concern. Cloud users usually encrypt files before moving it to the cloud. Unfortunately, deduplication and encryption are two conflicting technologies. While the aim of deduplication is to detect identical data segments and store them only once, the result of encryption is to make two identical data segments indistinguishable after being encrypted. This means that if data are encrypted by users in a standard way, the cloud storage provider cannot apply deduplication since two identical data segments will be different after encryption. On the other hand, if data are not encrypted by users, confidentiality cannot be guaranteed and data are not protected against attackers [3].

Convergent encryption technique is used to implement deduplication in the cloud environment. Convergent encryption, also known as content hash keying, is a cryptosystem that produces identical ciphertext from identical plaintext files [4]. But it introduces several security treats, like "confirmation of a file attack" and "learn the remaining information attack". The only effective approach to mitigate these attacks is to encrypt the contents of files with a non-convergent encryption scheme before storing.

This paper presents a system that implements a block level deduplication in the cloud storage. Additional encryption layer is added, with the help of a server component, to prevent well-known attacks against convergent encryption and thus protect the confidentiality of the data. Key management is also a serious issue in deduplication. Deduplication manager component in the architecture implements deduplication and efficiently manage the keys.

2. Related Work

MihirBellareproposes DupLess[4] Server aided encryption for deduplicated storage for cloud storage service provider.Message lock encryption is used to encrypt the file. Key Server (KS) is implemented to prevent brute-force attack.KS also derive keys, instead of setting keys to be hashes of messages.Authors of work[5] propose a cryptographically secure and efficient scheme, called a provable ownership of the file (POF), for a client to prove to the server that it indeed has the file.Twin Cloud [6] proposes architecture for secure outsourcing of data and arbitrary computations to an untrusted commodity cloud. The resource-constrained Trusted Cloudis used for pre-computations.Security criticaloperations are performed by the Trusted Cloud in the Setup Phase,whereas performance-critical operations are performed on encrypted data in parallelby the Commodity Cloud in the Query Phase.

Dekey [7] architecture proposes efficient and reliable convergent key management through convergent key deduplication and secret sharing.Dekey supports both file-level and block level deduplications. Work [8] pointed out the potential risks of cross-user source based-deduplication.Work [8] analyze these threats and propose a simple mechanism that enables cross-user deduplication while greatly reducing the risk of data leakage.Work [9] presents a system that includes the public cloud and the private cloud and also hybrid cloud which is a combination of the both public cloud and private cloud.

3. Proposed System

The proposed scheme performs deduplication at block level. The scheme consists of four basic components: user performs convergent encryption on the uploaded file, server applies additional layer of encryption on the uploaded files, third component deduplication manager (DM), is in charge of the block level deduplication and key management operations. Fourth component, cloud service provider physically stores the data blocks.

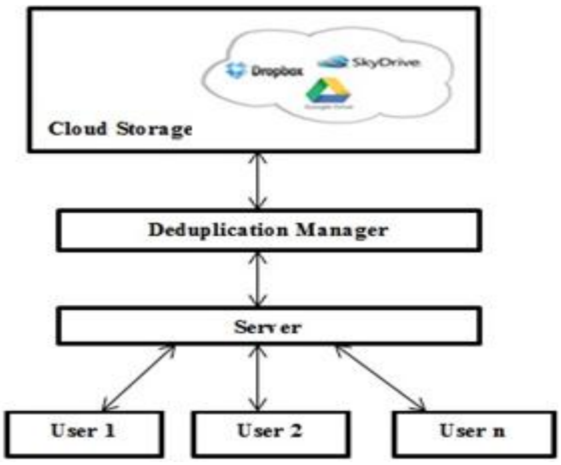

Figure 1: High Level Architecture

3.1 User

A user is any entity that wants to outsource data to the CSP and access the data later. In a deduplicated storage system, user only uploads unique data but does not upload any duplicate data to save the upload bandwidth. Each file is protected with the convergent encryption key to realize the authorized deduplication.

3.2 Server

Server authenticates users during the storage/retrieval phase. It encrypts/decrypts data uploaded by users in order to prevent attacks based on CE weaknesses. Server also verifies the signature of each file with the user"s public key during the retrieval phase.

3.3 Deduplication Manager (DM)

DM handles block level deduplication and stores metadata which include file signatures, hash values and pointers to the actual storage.DM also checks whether a user is authorized to retrieve a file that he/she has requested. DM communicates with cloud service provider (CSP) to store and retrieve data blocks.

3.4 Cloud Service Provider (CSP)

The role of CSP is to physically store data blocks. CSP is not aware of the deduplication and ignores any existing relation between two or more blocks. It is possible to make use of well-known cloud storage providers such as Google Drive, Amazon S3 and Drop box.

DM maintains a small database in order to keep track of file ownerships and to keep metadata generated during deduplication check. The tables used for this purpose are file table, pointer table, signature table and hash table.

- File table - File table contains the file id and user id fields.
- Pointer table - Pointer table contains the block id and the id of the block stored at the cloud storage provider.
- Signature table - Signature table contains the file id and the signature.
- Hash Table – Hash table contains hash code of each block B_i, corresponding pointer at the cloud storage and the number of users whose uploaded file contains that particular block.

4. Algorithm

4.1File Storage

During the storage procedure, users upload files to the system. Suppose user U_j wants to upload the file F1.

1. User
   a. Select file F1
   b. Calculate hash value (H_{F1}) of file F1
   c. Encrypt file using H_{F1} (convergent encryption)
   d. Sign the encrypted file F1
   e. Generate file identifier $F1_{id}$ by the concatenation of user id and file name

f. Create storage request, which consist of user id, file id, encrypted block and signature of file
g. Send the storage request to server

2. Server
   a. Receives a request from user U_j and authenticates U_j
   b. Encrypts file and signature using K_A (server"s secret key)
   c. Forward the new encrypted request to deduplication manager
3. Deduplication Manager
   a. DM receives the request from server
   b. Insert user id and file id fields to the file table
   c. Insert file id and signature to the signature table
   d. Splits the file(F1") into fixed sized blocks (B_i)
   e. For each block B_i compute hash value (K_i) and comparing it to the ones already stored in the hash table
   f. If search fails (new copy), send a new storage request to CSP and gets the pointer to the location at cloud storage
   g. If a match is found (duplicate copy), retrieves the pointer corresponding to the block from the hash table
   h. Update the pointer table
   i. Increment the number of owners of that block by one
   j. Repeat the procedure for all the blocks of the file F1"
4. CSP
   a. Receives storage request from DM
   b. Stores the block
   c. Return pointer

4.2 File Retrieval

During the retrieval procedure, a user asks to download a file from the system. Suppose user U_j wants to download the file F1.

1. User
   a. Send retrieval request to the server, request consist of user id and file id
2. Server
   a. Receives a request from user U_j and authenticates U_j
   b. If authentication does not fail, forward the request to deduplication manager
3. Deduplication Manager
   a. DM receives the request from server
   b. Proceeds only if U_j is the owner of the file
   c. Retrieves pointer from pointer table for all the blocks that composes the file
   d. Send request to the CSP and gets the blocks
   e. Merge blocks to the file F1"
   f. Build response with F1" and signature
   g. Send response to server
4. Server
   a. Receives response from DM
   b. Decrypts file F1" and signature using servers" secret key
   c. Perform signature verification
   d. If verification does not fail send the file F1' and signature to user
5. User
   a. Decrypt the file F1' using user"s secret key

4.3 File Delete

During the delete procedure, a user asks to delete a file from the system. Suppose user U_j wants to delete the file F1.

1. User
   a. Send delete request to the server, request is composed of user id and file id.
2. Server
   a. Receives a request from user U_j and authenticates U_j
   b. If authentication does not fail, forward the request to deduplication manager.
3. Deduplication Manager
   a. DM receives the request from server
   b. Proceeds only if U_j is the owner of the file
   c. Retrieves pointer from pointer table for all the blocks that composes the file
   d. For each block B_i checks the number of owners of the block
   e. If count is one send request to the CSP to delete the block
   f. Decrement the count value
   g. Remove all the entries corresponding to the block B_i from database
   h. Send confirmation message to server
4. Server
   a. Forwards the request to the user

4.4 File Update

Suppose a user U_j wants to modify the file F1. In this case U_j downloads the file F1, using the download procedure described above. U_j makes necessary changes to the file F1. After modification F1 becomes F2. Now U_j can upload F2 using the above described upload procedure. Block level deduplication concept is hidden from the user, so it is not necessary to inform other owners of the file F1 about the modification. Since F1 and F1' are two entirely different files, F1 remains in the cloud storage as it is before the modification and F1" will be stored as a new file.

5. Performance Analysis

A secure block level deduplication system has been implemented with the help of JAVA programming language and SQL database. Amazon S3 is used as cloud storage provider.

Different analysis proves that the overhead for block-level deduplication is affordable even with encryption. User performs convergent encryption on the selected file. DM splits the received encrypted file into blocks. A hash of each block is calculated in order to compare them to the ones already stored. This task is completely independent from the encryption technique used by clients. All the encryptions performed in the system do not affect the deduplication effectiveness since the encryption is deterministic. Therefore, the proposed system provides additional security properties without having an impact on the deduplication rate. The total cost of the storage and retrieval operation is linear for the encryption operations and almost linear for the lookup in tables therefore the metadata management is scalable.

5.1 Storage Space

The system will improve the memory usage while storing the file in cloud storage since block level deduplication points each duplicated blocks to existing data in cloud. Instead of checking throughout a file, block level deduplication will divide the file into certain blocks and identical blocks will be pointed to existing data stored on the cloud. This system will achieve the protection against COF and LRI attacks.

Deduplication check is performed on 10 different files with file size ranging from 10MB to 50 MB. Each file is divided into number of equal sized blocks and the size of each block is 5MB.

Table 1 shows the file size and storage space required at CSP without deduplication. The file stored as it is on the CSP. However, it does not check the duplicate contents while storing a file. All the files uploaded by the users will be stored at the cloud storage provider.

Table 1: Before Deduplication

User	File Name	Size (MB)	File at CSP
User1	Project.docx	10	10
User1	Accounts.docx	35	35
User1	cloud.docx	30	30
User1	Java.docx	20	20
User1	Projects.docx	40	40
User2	Projects.docx	50	50
User2	Accounts.docx	35	35
User2	Plan.docx	30	30
User3	Sample.docx	25	25
User3	Test.docx	25	25
Total Size			300

Table 2 describes the storage space required with deduplication. System will improve the memory usage while storing the file in cloud storage.

Table 2: After Deduplication

User	File Name	Size (MB)	File at CSP
User1	Project.docx	10	10
User1	Accounts.docx	35	35
User1	cloud.docx	30	30
User1	Java.docx	20	20
User1	Projects.docx	40	30
User2	Projects.docx	50	10
User2	Accounts.docx	35	0
User2	Plan.docx	30	30
User3	Sample.docx	25	25
User3	Test.docx	25	25
Total Size			215

Total space saved = 300-215 = 85MB. From the above two tables it is clear that deduplication process helps to reduce the storage space significantly. This is illustrated in the graph given below.

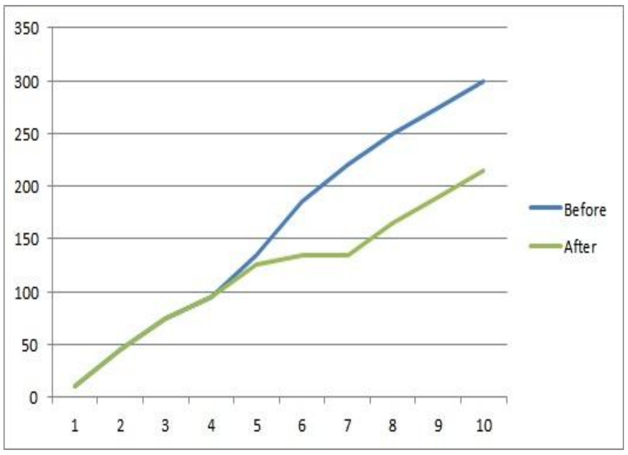

Figure 2: Storage Space Required at CSP Before and After Deduplication

5.2 Security

Consider the potential attack scenarios and possible issues that might arise. Assume that an attacker has full access to the storage. If the attacker has only access to the storage, he cannot get any information. File is encrypted with convergent encryption technique by the user and then further encrypted with server‟s secret key. Deduplication manager divides this file into blocks then stores the file in the cloud storage after deduplication check. Moreover, no metadata are stored at the cloud storage provider. Hence the attacker cannot perform any dictionary attack on predictable files. A worse scenario is the one in which the attacker manages to compromise the deduplication manager and thus has access to data and metadata. In this case, confidentiality and privacy would still be guaranteed since user retains the file keys and the signature of file is encrypted with server‟s secret key. The only information the attacker can get are data similarity and relationships between files, users and blocks. If the attacker compromises the server, only online attacks would be possible since this component directly communicates with users. The effect of such a breach is limited since data uploaded by users are encrypted with convergent encryption, which achieves confidentiality for unpredictable files.

6. Conclusion

The motivations that lead to the design of the proposed system, in a way that confidentiality and block-level deduplication are achieved at the same time, is explained. System is built on top of convergent encryption. Proposed system contains three core components: users, the server and the deduplication manager. Server is the core component which adds an additional layer of symmetric encryption. As the additional encryption is symmetric, the impact on performance is negligible. Proposed architecture prevents any single component from compromising the security of the whole system and prevents curious cloud storage providers from inferring the original content of stored data by observing access patterns or accessing metadata. The overhead of metadata management is minimal. It is fully compatible with standard storage APIs and transparent for the cloud storage provider, which does not have to be aware of the running deduplication system. Therefore, any potentially untrusted cloud storage provider such as Amazon, Dropbox and Google Drive, can play the role of storage provider.

The proposed system is implemented with the help of Java Programming language, SQLite database and Amazon Web Services. The basic operations storage, retrieval modification and deletion of files in cloud storage are discussed and results are analyzed. Experimental results show that deduplication implemented above the cloud storage service significantly reduces the storage space. Block-level deduplication helps to achieve greater reduction in storage space than file level deduplication. As part of future work, work may be extended with more security features such as proofs of retrievability, and search over encrypted data.

References

[1] https://en.wikipedia.org/wiki/Data_deduplication.
[2] http://searchstorage.techtarget.com/definition/data-deduplication.
[3] https://en.wikipedia.org/wiki/Cloud_storage
[4] Is Convergent Encryption really secure? http://bit.ly/Uf63yH.
[5] MihirBellare, SriramKeelveedhi, and Thomas Ristenpart. Dupless: Server-aided encryption for deduplicated storage. In USENIX Security Symposium, 2013
[6] Halevi, S., Harnik, D., Pinkas, B., Shulman-Peleg, A.: Proofs of ownership in remote storage systems. In: ACM CCS "11: ACM conference on Computer and communications security, pages 491-500, 2011
[7] 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.
[8] 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.
[9] D. Harnik, B. Pinkas, and A. Shulman-Peleg. Side channels in cloud services: Deduplication in cloud storage. IEEE Security and Privacy, 8(6):40–47, 2010.
[10] GauravKakariya andProf. SonaliRangdale. A hybrid cloud approach for secure authorized deduplication. International Journal of Computer Engineering and Applications, Volume VIII, Issue I, October 14.
[11] Daehee Kim, Baek-Young Choi, "HEDS: Hybrid deduplication approach for email servers", in Ubiquitous and Future Networks (ICUFN), Fourth International Conference, 2012, pp. 97-102.
[12] Pasquale Puzio. ClouDedup: Secure Deduplication with Encrypted Data for Cloud Storage,http://elastic-security.com/2013/12/10/cloudedup-secure-deduplication.
[13] Perttula. Attacks on Convergent Encryption.http://bit.ly/yQxyvl.
[14] D.Meister, "Advanced Data Deduplication Techniques and their Application", 2013.
[15] http://www.computerworld.com/article/2474479/data-center/data-deduplication-in-the-cloud-explained--part-one.html.
[16] http://www.druva.com/blog/a-simple-definition-what-is-data-deduplication/
[17] https://aws.amazon.com/s3/