Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 6, Issue. 6, June 2017, pg.73 – 79

Data Deduplication on Secured Cloud Storage

Pooja Umesh Kharade[1], Prof. Zaheed Shaikh[2]
[1]Department of Computer Engineering
[1]K.J. Somaiya College of Engineering, Mumbai, India
[2]Department of Computer Engineering
[2]K.J. Somaiya College of Engineering, Mumbai, India
[1] pooja.kharade@somaiya.edu; [2] zaheedshaikh@somaiya.edu

Abstract— Cloud storage is a remote storage service, where users can upload and download their data from anywhere and anytime. In the recent years, there has been a tremendous increase in the amount of digital data. This increased data occupies more storage space on cloud. Thus, the cloud storage server performs data compression i.e. data deduplication which eliminates the duplicate data and also saves the storage space. Also, cloud storage service poses challenges with respect to privacy and confidentiality of the data stored by the different users. We proposed a Data deduplication scheme, which can be used to eliminate the duplicate copies of data on cloud as well as preserves privacy and confidentiality of the data.
Keywords— Cloud storage, Data deduplication, confidentiality, privacy.

I. INTRODUCTION

Data deduplication is widely used data compression techniques. It is used to eliminate redundant copies of data stored on cloud. This saves the amount of storage space on the cloud. The data deduplication techniques also take into consideration the privacy and confidentiality of the stored data so that more number of users will be willing to store their data on cloud.

Literature survey is done on the various data deduplication techniques which shows that a lot of work is being carried out in this field. Data deduplication can be categorized into two different approaches: deduplication over unencrypted data and deduplication over encrypted data. Data deduplication is incompatible with the unencrypted data. Most of the schemes have been proposed to provide data encryption, while benefiting from a deduplication technique.

Many data deduplication schemes have been proposed in recent years such as hybrid cloud approach, cloudedup, message locked encryption. These techniques use the concept of convergent encryption which does not address the security issues completely. To cater this issue, we proposed a data deduplication scheme, which uses ECC algorithm for encryption and HMAC algorithm for checking duplicate copies of data.

II. RELATED WORKS

The following sections explain the survey of various papers regarding this concern. Different approaches that have been proposed for data deduplication are:

The hybrid cloud approach [1] aims at solving the problem of authorized data deduplication by using the differential privileges of users. A hybrid cloud architecture is considered in this approach which consists of a public and private cloud. Unlike existing data deduplication systems, the private cloud is involved as a proxy to allow data owner/users to securely perform duplicate check with differential privileges.
The system model of hybrid cloud approach consists of the following entities:

Storage cloud service provider (S-CSP): This entity provides data storage service in the private cloud. It stores data on the behalf of the users and outsources the data on demand of the users It eliminates the storage of duplicate data copies via deduplication.

Data users: A user is the entity who wants to outsource their data to the S-CSP and demand it later. In the authorized deduplication system, each user is issued a set of privileges in the setup of the system. Each file is protected with the convergent encryption key and privilege keys to realize the authorized deduplication with differential privileges.

Private cloud: This is new entity has been introduced into the system to facilitate user's secure usage of cloud storage. Specifically, since the computing resources at data user/owner side are restricted and the public cloud is not fully trusted in practice, private cloud is able to provide data user/ owner with an execution environment and infrastructure working as an interface between user and the public cloud. The private keys for the privileges are managed by the private cloud, who answers the file token requests from the users.

Convergent Encryption:
A user (or data owner) derives a convergent key from each original data copy and encrypts the data copy with the convergent key. In addition, the user also derives a tag for the data copy, such that the tag will be used to detect duplicates. It is assumed that the tag correctness property holds, i.e., if two data copies are the same, then their tags are the same. To detect duplicates, the user first sends the tag to the server side to check if the identical copy has been already stored. Both the convergent key and the tag will be stored on the server side and are independently derived, and the tag cannot be used to deduce the convergent key and compromise data confidentiality.

A convergent key consist of the following four primitive functions:

- KeyGen(M): K is the key generation algorithm that maps a data copy M to a convergent key K.
- Enc(K,M): C is the symmetric encryption algorithm that takes both the convergent key K and the data copy M as inputs and then output a ciphertext C.
- Dec(K,C): M is the decryption algorithm that takes both the convergent key K and the ciphertext C as inputs and then outputs the original data copy M.
- TagGen(M): T(M) is the tag generation algorithm that maps the original data copy M and outputs a tagT(M).

System Architecture:

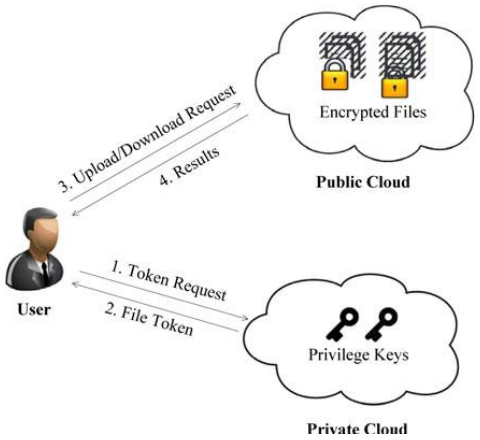

Fig.1 Hybrid Cloud Approach [1]

ClouDedup [2] provides the combined advantages of deduplication and convergent encryption by coping with the inherent security exposures of convergent encryption. The security of ClouDedup relies on its new architecture whereby in addition to the basic storage provider, a metadata manager and an additional server are defined: the server adds an additional encryption layer to prevent well-known attacks against convergent encryption and thus protect the confidentiality of the data; on the other hand, the metadata manager is responsible of the key management task since block-level deduplication requires the memorization of a huge number of keys. Therefore, the underlying deduplication is performed at block-level and we define an efficient key management mechanism to avoid users to store one key per block.

The scheme consists of two basic components: a server that is in charge of access control and that achieves the main protection against COF and LRI attacks; another component, named as metadata manager (MM), is in charge of the actual deduplication and key management operations.

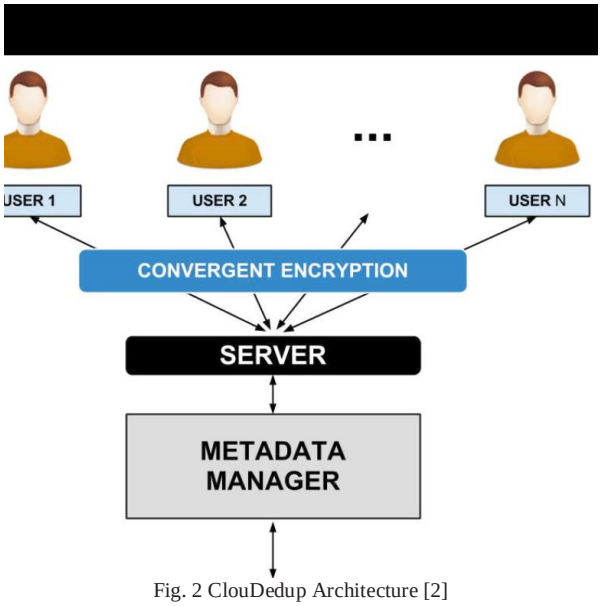

Fig. 2 ClouDedup Architecture [2]

User:
The role of the user is limited to splitting files into blocks, encrypting them with the convergent encryption technique, signing the resulting encrypted blocks and creating the storage request. In addition, the user also encrypts each key derived from the corresponding block with the previous one and his secret key in order to outsource the keying material as well and thus only store the key derived from the first block and the file identifier.

Server:
The server has three main roles: authenticating users during the storage/retrieval request, performing access control by verifying block signatures embedded in the data, encrypting/ decrypting data traveling from users to the cloud and viceversa. The server takes care of adding an additional layer of encryption to the data (blocks, keys and signatures) uploaded by users. Before being forwarded to MM, data are further encrypted in order to prevent MM and any other component from performing dictionary attacks and exploiting the well-known weaknesses of convergent encryption. During file retrieval, blocks are decrypted and the server verifies the signature of each block with the user's public key. If the verification process fails, blocks are not delivered to the requesting user.

Metadata Manager(MM):
MM is the component responsible for storing metadata, which include encrypted keys and block signatures, and handling deduplication. Indeed, MM maintains a linked list and a small database in order to keep track of file ownerships, file composition and avoid the storage of multiple copies of the same data segments.

The tables used by MM are structured as follows:
• File table. The file table contains the file id, file name, user id and the id of the first data block.

• Pointer table. The pointer table contains the block id and the id of the block stored at the cloud storage provider.
• Signature table. The signature table contains the block id, the file id and the signature.

Cloud Service Provider:
SP is the most simple component of the system. The only role of SP is to physically store data blocks. SP is not aware of the deduplication and ignores any existing relation between two or more blocks. Indeed, SP does not know which file(s) a block is part of or if two blocks are part of the same file.

DeDu [3] is a deduplication storage system consists of two major components, a front-end deduplication application and Hadoop Distributed File System. Hadoop Distributed File System is common back-end distribution file system, which is used with a Hadoop database. With the deduplication applications, a scalable and parallel deduplicated cloud storage system can be effectively built up. Further VMware is used to generate a simulated cloud environment.

The most prominent manifestation of MLE is Convergent Encryption (CE), introduced by Douceur et al. [4]CE was used within a wide variety of commercial and research storage service systems. Letting M be a file's data, a client first computes a key K by applying a cryptographic hash function H to M, and then computes ciphertext C via a deterministic symmetric encryption scheme. A second client B encrypting the same file M will produce the same C, enabling deduplication.

Bellare et al. proposed DupLESS [5] that provides secure deduplicated storage to resist brute force attacks. In Dup-LESS, a group of affiliated clients (e.g., company employees) encrypt their data with the aid of a Key Server (KS) that is separate from a Storage Service (SS). Clients authenticate themselves to the KS, but do not leak any information about their data to it. As long as the KS remains inaccessible to attackers, high security can be ensured. Obviously, Dup-LESS cannot control data access of other data users in a flexible way. Alternatively, a policy-based deduplication proxy schemes was proposed but it did not consider duplicated data management and did not evaluate scheme performance.

Limitations of Convergent Encryption

Convergent encryption suffers from some weaknesses which have been widely discussed in the literature [2]. As the encryption key depends on the value of the plaintext, an attacker who has gained access to the storage can perpetrate the so called "dictionary attacks" by comparing the ciphertexts resulting from the encryption of well-known plaintext values from a dictionary with the stored ciphertexts. Indeed, even if encryption keys are encrypted with users' private keys and stored somewhere else, the potentially malicious cloud provider, who has no access to the encryption key but has access to the encrypted chunks (blocks), can easily perform offline dictionary attacks and discover predictable files. This issue arises in where chunks are stored at the storage provider after being encrypted with convergent encryption.

The following two attacks are possible against convergent encryption: confirmation of a file (COF) and learn-the-remaining-information (LRI). These attacks exploit the deterministic relationship between the plaintext and the encryption key in order to check if a given plaintext has already been stored or not. In COF, an attacker who already knows the full plaintext of a file, can check if a copy of that file has already been stored. If the attacker is the cloud provider or an insider, he might also learn which users are the owners of that file. Depending on the content of the file, this type of information leakage can be dangerous. Hence, a strategy is needed to achieve a higher security degree while preserving combined advantages of both convergent encryption and deduplication.

III. PROPOSED WORK

The proposed framework of data deduplication technique uses ECC algorithm for encryption and decryption of the data and HMAC algorithm for calculating the hash tag of the data for checking for duplicate copies.

A. ECC Algorithm

Elliptic curve cryptography (**ECC**) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-ECC cryptography (based on plain Galois fields) to provide equivalent security. Elliptic curves are applicable for encryption, digital signatures, pseudo-random generators and other tasks.

The primary benefit promised by elliptic curve cryptography is a smaller key size, reducing storage and transmission requirements, i.e. that an elliptic curve group could provide the same level of security afforded by an RSA-based system with a large modulus and correspondingly larger key: for example, a 256-bit elliptic curve public key should provide comparable security to a 3072-bit RSA public key.
Elliptic curve cryptosystems (ECCs) include key distribution, encryption algorithms. The key distribution algorithm is used to share a secret key and the encryption algorithm enables confidential communication.
The equation of an elliptic curve is given as,

$$y^2 = x^3 + ax + b$$

Few terms that will be used,

E -> Elliptic Curve
P -> Point on the curve
n -> Maximum limit (This should be a prime number)

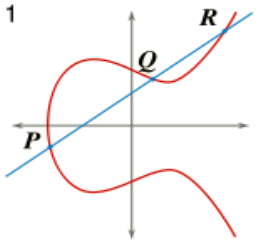

Fig. 3 Elliptic Curve

Key Generation:
Key generation is an important part where we have to generate both public key and private key. The file will be encrypted and decrypted using the secret key (combination of public key and private key).

- Now, we have to select a number **'d'** within the range of **'n'**.
- Using the following equation we can generate the public key

Q=d*P

d = The random number that we have selected within the range of (**1 to n-1**).
P is the point on the curve.

'Q' is the public key and **'d' is the private key.**

B. HMAC

In cryptography, a **keyed-hash message authentication code** (**HMAC**) is a specific type of message authentication code (MAC) involving a cryptographic hash function and a secret cryptographic key. It may be used to simultaneously verify both the data integrity and the authentication of a message, as with any MAC. Any cryptographic hash function, such as MD5 or SHA-1, may be used in the calculation of an HMAC; the resulting MAC algorithm is termed HMAC-MD5 or HMAC-SHA1 accordingly. The cryptographic strength of the HMAC depends upon the cryptographic strength of the underlying hash function, the size of its hash output, and on the size and quality of the key .

The current HMAC specification which is defined as **H**(key ‖ **H**(key ‖ message)) because the outer application of the hash function masks the intermediate result of the internal hash. The values of ipad and opad are not critical to the security of the algorithm, but were defined in such a way to have a large Hamming distance from each other and so the inner and outer keys will have fewer bits in common.

HMAC(K,m)=H((K' XOR opad) || H(K' XOR ipad) || m)

where

H is a cryptographic hash function,

K is the secret key,

m is the message to be authenticated,

K' is another secret key, derived from the original key K (by padding K to the right with extra zeroes to the input block size of the hash function, or by hashing K if it is longer than that block size),

|| denotes concatenation,

⊕ denotes exclusive or (XOR),

opad is the outer padding (0x5c5c5c…5c5c, one-block-long hexadecimal constant),

and ipad is the inner padding (0x363636…3636, one-block long hexadecimal constant).

C. System Architecture

The proposed system consists of three entities: the user, the server and the cloud service provider. The user is the entity which wants to store its data in the cloud storage. The server is the entity where all the processing of the data and the calculations takes place. The CSP provides the data storage facility. Along with encrypted data, it also stores the user related data (parameters: tokens, public key, private key, system hash key, random number).

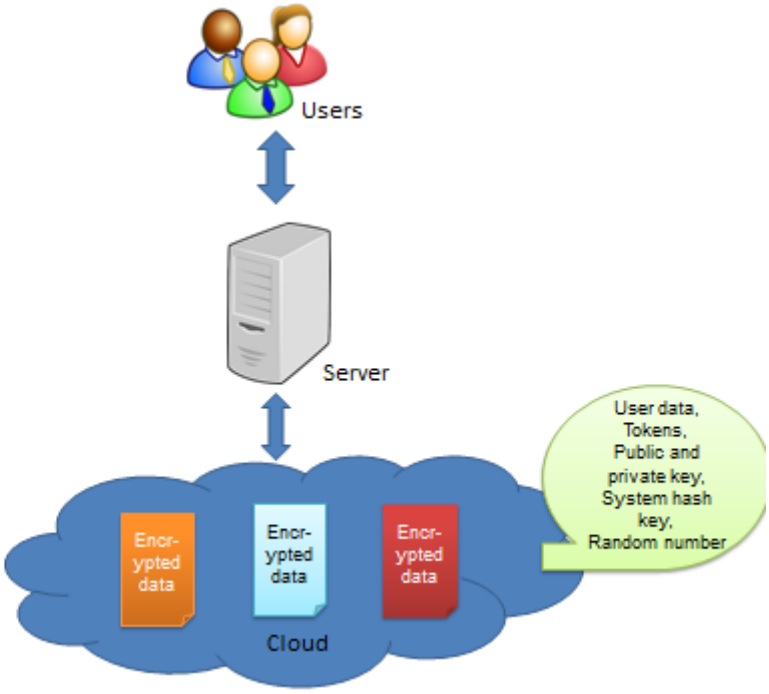

Fig. 4 System Architecture

The user first registers with the system. A pass phrase is mailed to the user which is used for authentication during uploading files. Using ECC algorithm a public and private key is generated and is stored in the cloud database. A secret key which is a combination of public and private key is used for encryption and decryption. A user hash key is also mailed to the user. This user hash key is generated as system hash key XOR random number(r). This random number is unique for all the users whereas system hash key is common for all the users. The system hash key is used as the key for calculating hash tag using HMAC-SHA256 algorithm. The user calculates the hash tag and sends it to the CSP. The CSP checks whether such hash tag already exits.

- If the hah tag already exists, the data is not uploaded again, instead a link is stored with the id of the user who has already uploaded the same file earlier.
- If hash tag doesn't exists, the file is encrypted using the secret key and then stored in the cloud database.

The system also focuses on the authorized authentication. Thus an one-time password (OTP) is generated every time when the user wants to upload/download files. This OTP is sent to the user on his/her registered email id. The user has to enter this OTP for token validation every time. Also the user is also asked to enter the user hash key for uploading/downloading the file. This user hash key is XORed with the stored random number which is unique for every user. The file is uploaded or downloaded only if the the XORed value is equal to the system hash key.

IV. RESULT

A prototype for the proposed work has been implemented using Apache Tomcat server and Xampp server and has been deployed on Ulteo Cloud. During the experimentation it is found that the duplicate files are not uploaded again, instead only a link is stored with the id of the user who has already uploaded the same file earlier, thus saving the cloud storage space.

CONCLUSION

By using ECC algorithm for data encryption we make an attempt to eliminate the security issues of convergent encryption. As data encryption and decryption by ECC algorithm requires combination (secret key) of two keys (private and public key) which are stored in cloud, the privacy and confidentiality of the data is achieved.

REFERENCES

[1] Jin Li, Yan Kit Li, Xiaofeng Chen, Patrick P.C. Lee, and Wenjing Lou, "A Hybrid Cloud Approach for Secure Authorized Deduplication," IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 26, NO. 5, MAY 2015.
[2] P. Puzio, R. Molva, M. Onen, and S. Loureiro, "ClouDedup: Secure deduplication with encrypted data for cloud storage," in Proc. IEEE Int. Cof. Cloud Comput. Technol. Sci., 2013, pp. 363–370, doi:10.1109/CloudCom.2013.54.
[3] Z. Sun, J. Shen, and J. M. Yong, "DeDu: Building a deduplication storage system over cloud computing," in Proc. IEEE Int. Conf. Comput. Supported Cooperative Work Des., 2011, pp. 348–355, doi:10.1109/CSCWD.2011.5960097.
[4] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in Proc. Cryptology—EUROCRYPT, 2013, pp. 296–312, doi:10.1007/978-3-642-38348-9_18.
[5] M. Bellare, S. Keelveedhi, and T. Ristenpart, "DupLESS: Server aided encryption for deduplicated storage," in Proc. 22nd USENIX Conf. Secur., 2013, pp. 179–194.
[6] C. Fan, S. Y. Huang, and W. C. Hsu, "Hybrid data deduplication in cloud environment," in Proc. Int. Conf. Inf. Secur. Intell. Control, 2012, pp. 174–177, doi:10.1109/ISIC.2012.6449734.
[7] 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. Parallel Distrib. Syst., http:// doi.ieeecomputersociety.org/10.1109/TPDS.2013.284, 2013.
[8] 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.
[9] https://en.wikipedia.org/wiki/Elliptic_curve_cryptography
[10] https://en.wikipedia.org/wiki/Hash-based_message_authentication_code
[11] https://en.wikipedia.org/wiki/Ulteo_Open_Virtual_Desktop
[12] Is Convergent Encryption really secure? http://bit.ly/Uf63yH.
[13] Perttula. Attacks on convergent encryption. http://bit.ly/yQxyvl.
[14] John Pettitt. Hash of plaintext as key? http://cypherpunks.venona.com/date/1996/02/msg02013.html.
[15] Dropbox, A file-storage and sharing service. (2016). [Online]. Available: http://www.dropbox.com
[16] Google Drive. (2016). [Online]. Available: http://drive.google.com
[17] Mozy, Mozy: A File-storage and Sharing Service. (2016). [Online]. Available: http://mozy.com/

