IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 6, Ver. VI (Nov.-Dec. 2016), PP 12-18
www.iosrjournals.org

Implementation of new Secure Mechanism for Data Deduplication in Hybrid Cloud

Dr. K. Gangadhara Rao[1], Dr.B.Basaveswara Rao[2], Qutaiba Mumtaz Dawood[3]
[1,3]*Department of Computer Science and Engineering,*
[2]*Computer Centre, Acharya Nagarjuna University, Guntur 522501, Andhra Pradesh, INDIA.*

Abstract: *Cloud computing provides huge amount of area for storage of the data, but with an increase of number of users and size of their data, cloud storage environment faces earnest problem such as saving storage space, managing this large data, security and privacy of data. To save space in cloud storage one of the important methods is data deduplication, it is one of the compression technique that allows only one copy of the data to be saved and eliminate the extra copies. To offer security and privacy of the sensitive data while supporting the deduplication, In this work attacks that exploit the hybrid cloud deduplication have been identified, allowing an attacker to gain access to the files of other users based on very small hash signatures of these files. More specifically, an attacker who knows the hash signature of a file can convince the storage service that he/she owns that file, hence the server lets the attacker to download the entire file. To overcome such attacks,the hash signature is encrypted with the user password. As a proof of concept a prototype of the proposed authorized deduplicate is implemented and conducted the test bed experiments using the prototype. Performance measurements indicate that the proposed Deduplication system incurs minimal overhead in the context of uploading, bandwidth compared to native deduplication.*

Keyword: *Deduplication, duplication authorized check, hybrid cloud.*

I. Introduction

Cloud computing offers computing services and storage services.Day after day cloud computing becomes more popular as the number of users increase. There are many companies which are delivering services from the cloud, for example, Google –has the private cloud which is delivering services to users such as email access, documents application, text translations, maps, Google drive, Microsoft Skydrive and many more.These companies offering services via the cloud with high availability of storage and massively parallel computing at low cost. As cloud computing becomes popular, voluminous data gets stored in the cloud which lead to one big challenge is that how to save storage and manage the data in secured fashion simultaneously maintaining privacy. As per IDC analysis report the volume of data in the world would be 40 trillion gigabytes in 2020[19]. Deduplication is a technique for eliminating duplicate copies of repeating data in the cloud storage. By deduplication the cloud service provider stores only one copy of the file and passing the link on request .Deduplication can be applied either File level or block level. For File –level, it deletes the duplicate copy of the same file (identical file). At the block level delete duplicate block occurs in non-identical file. Providers of cloud-based storage such as Dropbox [1], Google Drive [2], and Mozy [3] can save on storage costs via deduplication: should client(s) upload the same file, the service detects this and stores only a single copy. Unfortunately, this action of deduplication would lead to a number of threats potentially affecting the storage system [5][6]. Although the deduplication technique has a lot of advantages, security and privacy of the sensitive data is challenging by both inside and outside attackers. To increase security level and privacy, users need to encrypt their data before storing on a remote server, use of the standard encryption method with different keys for each user data which lead different ciphertexts to the same data so cloud storage will get different ciphertexts for identical file which making duplicate check in cloud storage impossible. To solve this problem convergent encryption [10] has been proposed to ensure data privacy while making deduplication feasible. In convergent encryption algorithm key for encrypting the data is derived from the data itself rather than a random cryptographic key to produce the same ciphertext for each identical file of the data owner. For example user derived the encryption key from the message M such that K=H(M) where H is cryptographic hash function ,so here C=E(K,M)=E(H(M),M),with this simple example user can get the same ciphertext for the same file so cloud storage provider can check duplicate of ciphertext without any knowledge of the original data. From this, one can say that convergent encryption seems to be good candidate for the adoption of encryption and deduplication in the cloud storage environment. To perform the duplicate check for some file, the user needs to get the file token from the private cloud server by sending tag and user ID generated by MD5 hashing technique. The private cloud server will generate token and encrypt it with the user password to increase the security level and then send the encrypted token to the user. The authorized duplicate check for this file can be performed by the user with public cloud by sending the token to public cloud which is going to match the token

with hash table. Based on the results of duplicate check, the user either uploads this file or runs POW. Shai Halevi [6] pointed out the weakness of the security in traditional deduplication systems with only a short hashing value. To overcome this security issue, they presented the concept of proof of ownership (POW), which is an interactive algorithm installed in cloud server that enables users to prove their ownership of data copies. The cloud storage derives a short value ϕ (M) from a data copy M. To prove the ownership of the data copy M, the user needs to send ϕ'_F to the cloud storage such that $\phi'_F = \phi$ (M). After user proof ownership of stored file in cloud storage, S-CSP will allow user to download encrypted file that the user can decrypt with convergent key that is saved locally. Each file uploaded to the cloud storage is bounded by a set of privileges to limit the access to the stored files and specify which kind of user can access and perform the duplicate check, if the user has matched copy and a set of privileges then the user is able to find and download the duplicate copy.

1.1 Objective

In this paper, an attempt has been made to implement a novel secure mechanism for data deduplication in hybrid cloud. The sensitive data of the user is protected through the Symmetric encryption and Convergent encryption mechanisms, which is also compatible with the hybrid storage systems. In a more detailed ways, the user generates the hash value as File Tag and then sends Token request along with Tag and User ID to the private cloud, The private cloud loads the associated privilege keys of the user and generates the token, to ensure higher security, we encrypt the token with user Id, the User sends duplicate check request to public cloud along with Token received from private cloud. The public cloud matches the Token file with existing Hash table. If a file duplicate is found, the user needs to run the POW protocol with the public cloud to prove the file ownership. If no duplicate is found, proof from the public cloud will be returned, the user encrypt file with the convergent encryption and compressed with LZ4 and upload with the (user ID, File, Token).

To recover data copies, The User sends the file name to the public cloud server, the public cloud server checks whether the user is eligible to download the file. In other words, the data can only be accessed by the by the authorized users ie.who own the corresponding data copy. Security analysis demonstrates that the proposed deduplication systems are secure in terms of the definitions specified in the proposed security model. Specifically, the user without corresponding privileges cannot perform the duplicate check. Furthermore, unauthorized users cannot decrypt the ciphertext even if they collude with the public cloud. The deduplication system proposed is implemented and the overhead is evaluated, it is verified that the overhead is minimal compared to the normal convergent encryption and file upload operations.Looking at the comparison (table 1).

Table 1

System proposed by Jin Li. et al[4]	Proposed system
1. The user creates a hash for each file by using the Sha-1 algorithm 2. The private cloud generates unique token for each file by using HMAC-SHA1. 3. There is no encryption for the token from the side of private cloud. 4. There is no compression of the file from the user's side.	1. The user creates a hash for each file by using the MD-5 algorithm 2. The private cloud generates unique token for each file by using HMAC-MD5. 3. The token gets encrypted with user ID by the private cloud and the same would be send back to the user. The user forwards the token to the public cloud to check whether the file is duplicated or not. 4. At the user level, the user compresses the file before sending to the public cloud for final store, by using LZ4 algorithm.

1.2 Organization

The organization of the paper is as follows. Section 2 presents literature review. Section 3, discusses the preliminaries of Deduplication. In Section 4 presents the proposed System architecture for secure deduplication. The security analysis is discussed in Section 5. The Methodology and Implementation of the same is presented in section 6. Section 7 evaluates the performance of the proposed Secure Deduplication system. The Conclusion and Future work are presented in Section 8.

II. Literature Review

Mihir Bellare et al. [8], proposed an architecture that provides secure deduplicated storage which resists brute-force attacks, and they have realized the same in a system called DupLESS. Dupless is used to provide secure deduplicated storage. For cloud storage service provider to save space by only storing one copy of each file uploaded .Message lock encryption provides a way to achieve secure deduplication, which has been the target of several cloud service providers. The authors have shown that their work is a primitive of both theoretical and practical interest [9].Yuan et al. J. Li [13] addressed the key-management issue by proposing a scheme called Dekey, an efficient reliable convergent key management scheme for secure deduplication.Sven Bugiel et al.[11] proposed a novel architecture and protocols that accumulate slow secure computations over time and provide the possibility to query them in parallel on demand by leveraging the benefits of Cloud

computing. Zhang et al. [16] also introduced the hybrid cloud methods to support security aware data intensive computing. The work considers pointing the authorized deduplication issue over information in public cloud. J. Xu et al. [14] showed a secure convergent encryption for efficient encryption, without considering issues of the key management and block level deduplication. There are several implementations of convergent encryption for secure deduplication [17] [18]. Proofs of Ownership in Remote Storage Systems [6], they have offered a solution based on specific encoding and Merkle-Trees and measured the performance by comparing with client side deduplication. [15] extended POW for encrypted files, but they do not show how to reduce the key management overhead. In POWs term user proves to server that it in fact holds the whole data of the file rather than just short information about it. [12] Proposed a constant cost deduplication schema, which enables the deduplication of both the files and their corresponding authentication tags. Jin Li et al. [4] proposed an authorized deduplication to protect the data security by including differential privileges of users in the duplicate check. The drawbacks in their schema include Token security and bandwidth consumption. They didn't apply any kind of encryption technique for the token .Their schema takes more bandwidth because of the reason that they never applied any compression technique to reduce the size of the file**.** In the proposed system higher priority is given to the security of the Token, the Token is encrypted in the private cloud with User ID and share the key with public cloud this step ensure that the Token will be safe during transfer from one place to another , and also applying LZW compression algorithm to reduce file size before sending it to Public cloud storage provider. MD5 is used as the hashing technique rather than SHA-1 hashing technique in proposed system, which takes 32 bits less in than its counterpart SHA-1 when the length of the message digest is considered.

III. Preliminaries

This part of the paper first defines the notations used in this paper, followed by a brief review of some security primitives utilized as a part of our secure deduplication. The notations used are given in Table 2.

Table 2

Acronym	Description
S-CSP	Storage –Cloud Service provider.
POW	Proof Of ownership.
K_F	Convergent encryption key for file F.
C_F	Encrypted File.
P_U	Privilege set of a User U.
ϕ'_F	Token of file F.
H	Hash function.
AES	Advanced Encryption Standard.
CBC	Cipher Block Chaining.
HMAC	Hash Message Authentication Code
LZW	Lempel-Ziv-Welch

Notations Used in This Paper

3.1 Symmetric encryption. Symmetric encryption uses same key K for both encryptions of plaintext and decryption of ciphertext. A symmetric encryption consists of three primitive functions:

- KeyGen ($1^{\square}$) $\rightarrow$ K is the key generation algorithm that generates K using security parameter $1^{\square}$;
- Enc_{SE} (K, M) $\rightarrow$ C is the symmetric encryption algorithm that takes the secret K and message M and then outputs the ciphertext C; and
- Dec_{SE} (K, C)$\rightarrow$ M is the symmetric decryption algorithm that takes the secret K and ciphertext C and then outputs the original message M.

3.2 Convergent encryption. Traditional encryption though provides data confidentiality, is incompatible with deduplication.Because of this reason that the traditional encryption requires different users to encrypt their data with their own keys. In this case where two identical files being encrypted with different keys will give two different ciphertexts so deduplication will be impossible. To overcome this problem convergent encryption is used, the main idea of convergent encryption is that each user encrypts his data by using a convergent key K with symmetric encryption. The convergent key is derived from the copy itself by using some hash functions, the user computes the encrypted file C_F = Enc (K_F, F), the key K_F = KeyGen (F). In addition to this the user also derives a tag for the data copy such that the tag will be used to detect duplicates. If two data copies are same then their tags are also same. The convergent key K and the tag are independently derived, and one cannot use the tag to deduce convergent key. To check duplications the user sends the tag of the data that wish to upload to the cloud storage, and S-CSP will check whether the data is already stored or not. The Convergent encryption scheme can be defined with four primitives functions:

- $KeyGen_{CE}$ (M) → K is the key generation algorithm that maps a data owner M to a convergent key K;
- Enc_{CE} (K, M) → C is the symmetric encryption algorithm that takes both the data owner M and the convergent key K as inputs and then outputs a ciphertext C;
- Dec_{CE} (K, C) → M is the decryption algorithm that takes both the convergent key and the cipthertext C as inputs and then outputs the original data owner M; and
- TagGen (M) → T (M) is the tag generation algorithm that maps the data owner M and outputs a tag T(M).

3.3 Proof of ownership. [6]The idea of proof of ownership enables user to prove their ownership of data copies to the server. The storage server derives a short value ϕ (M) from a data copy M .To prove the ownership of the data copy user needs to send ϕ′ to storage server such that ϕ′=ϕ (M).

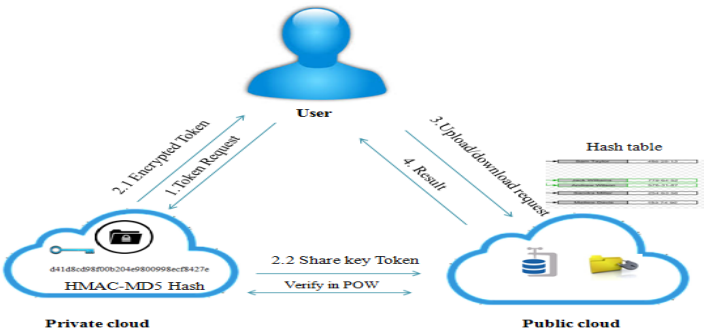

Figure.1. Architechure for secure deduplication

IV. System Architecture For Secure Deduplication

There are three components in the proposed system, *user*, *private* cloud, and *public cloud* as shown in figure 1.

- Public cloud provider is a component that provides a data storage services, to reduce storage cost the cloud storage eliminates the redundant data via deduplication technique and keeping only a single copy of data, a public cloud provider performs deduplication check with the help of the token received from the user combining this with the hash table look up if the contents of two files uploaded by the same user are same then stored only one of them.
- A user is a component that wants to outsource data storage and access the data later and have access to the private cloud which will generate file tokens after the user requested. The role of the user is to generate file tag by using MD-5 algorithm and compress, encrypt the file before uploading.
- Private cloud provides an execution environment and infrastructure for the user and works as an interface between the public cloud and user. The interface offered by the private cloud enable the user to securely perform the store and compute operations.

V. Security Analysis

This section discusses the security analysis for the deduplication system. Some basic cryptographic tools have been applied into our construction to achieve secure deduplication. To show the security of this protocol, it is assumed that the underlying building blocks are secure, including the POW scheme. Thus, the security will be analyzed based on the above security assumptions. In this system, the public cloud assumed to follow the protocols. If the data file has been successfully uploaded and stored at servers, then the user who owns the file can convince the servers based on the correctness of the POW. The file will be recovered by the user with correct privileges key. The confidentiality against two types of adversaries is considered here.The first type of adversary is defined as dishonest users who aim to retrieve files stored at the storage cloud service provider they do not own. The second type of adversary is defined as a group of storage cloud service provider and users. Their goal is to get the useful information of file content they do not own individually by launching the collusion attack. The attacks launched by these two types of adversaries are denoted by Type-I attack and Type-II attack, respectively.

Type-I

This type of adversary tries to convince the storage cloud service provider with some auxiliary information to get the content of the file stored in the storage cloud service provider. To get the file stored in the storage cloud service provider, the user needs to perform a correct POW protocol. In this way, if the adversary wants to get the file, he must has a short value ϕ′ , the attacker cannot get the auxiliary value used to perform POW if he does not own the file. Thus, based on the security of POW, the security against a Type-I attack is easily derived.

Type-II

The second type of adversary try to get the useful information of file content by a group of users/csps which they do not own individually by launching the collusion attack. In our system the convergent key is stored by user locally, this means the confidentiality of the data stored at the storage cloud service provider is guaranteed even if some storage cloud service provider collude.

VI. Methodology and Implementation

This system is divided into two sections first one is for upload file, and the second one is for the download file:

6.1 Methodology for File Upload:

1. Register user data on the private cloud server.
2. User selects the file to upload and generate the hash value as File Tag by using a MD-5 algorithm.
3. User sends Token request along with Tag and User ID to the private cloud.
4. The private cloud loads the associated privilege keys of the user and generates the token with HMAC-MD-5 then encrypts the token with user password and send to the user.
5. User sends duplicate check request to public cloud along with Token received from private cloud.
6. In deduplication check, it matches the Token file with existing Hash table, two cases are there:

- Case1:If a file is found to be duplicate, the user needs to run the POW protocol with the public cloud to prove the file ownership, if it is passed the user will be provided a pointer to the file. The user sends the privilege set for the file as well as the proof to the private cloud server after receiving the request, the private cloud server first verifies the proof from public cloud. If it is passed, the private cloud computes the tag and send to the user which allows him to download the file.
- Case2: If a file is found to be unique, proof from the public cloud will be returned, the user will send the privilege set for the file as well as the proof to the private cloud after receiving the request, the private cloud server first verifies the proof from public cloud. If it is passed, the private cloud server computes the tag and sends it back to the user. Finally, the user encrypts the file with the convergent encryption and compresses the file with LZW and upload the file, the token and the user ID to the SCSP.

6.2 Methodology for File downloading:

1. The User sends the file name to the public cloud server.
2. The public cloud server checks whether the user is eligible to download the file.
3. If failed, the public cloud server sends back an abort signal to the user to indicate the download failure.
4. If it is passed, the public cloud server returns the corresponding compressed ciphertext.
5. The user decompresses the file and then decrypts the same using the convergent key that is stored locally.

6.3 Implementation:

The implementation of the proposed deduplication system consists of three components. The three components are implemented as Java applications, and run on computer equipped with a 2.4 GHz Pentium Dual-Core CPU and 4GB RAM. The test machine runs 64 bit version of windows 7**.** A *client* program is used to model the data users to carry out the file upload process. A *private server* program is used to model the private cloud which manages the file token computation and private keys. A *storage server* program is used to model the cloud storage provider which stores and deduplicates the file.

The *client* program, provides the following functions

- FileTag (File) –It computes MD-5 hash of the File as File Tag.
- TokenReq (Tag,UserID) –It requests the Private cloud for File token generation.
- DupcheckReq (Token) –It request the public cloud for duplicate check of the file.
- ShareTokenReq (Tag, priv) –It requests the private cloud to generate the share file token with the file tag and sharing privilege set.
- FileCompression (File) –It compresses the file using a LZW algorithm.
- FileEncrypt (C_{File}) –It encrypts the compressed file with convergent encryption using 256-bit AES algorithm in (CBC) mode, where the convergent key is from MD-5 hashing of the file.
- FileuploadReq (FileID, CFile, Token) –It uploads the compressed file data to the public cloud if its unique and update the hash table.

In the *Private server* program, we provide the following functions:

- TokenGen (Tag, UserID) –It loads the privilege keys of the user and generate the Token with HMAC-MD-5 algorithm.
- TokenEnc (Token) - It encrypts the Token with user ID.
- ShareToken(Tag,priv) –It generates the share token with privilege key.s
- Sharekey (key) –private cloud shares the key with public cloud for decryption.

In the *public server* program, we provide the following functions:

- TokenDec (Token)- It decrypts the token using key received from Private cloud.

- DupCheck (Token) –It searches the file token to the hash table for duplicate by using linear search algorithm.
- FileStore (FileID, File, Token) –It stores the file on the public cloud and update the hash table.

VII. Performance Evaluation and Results

The performance evaluation of the proposed secure deduplication system, is a two step process.

8.1 Effect of File Size:

To evaluate the effect of file size in the system, 50 unique files of different size (10-400) MB are uploaded. To show the effect of file size in the different steps upload files are divided into five groups, each group contains 10 files, in the grope_1 the file size start (10-50) MB, group_2 (50-100),group_3 (100-200) MB, group_4 (200-300) MB, group_5 (300-400) MB, the uploading process is a seven step mechanism 1) Tagging 2) Token Generation 3) Duplicate Check 4) Share Token Generation 5) File Encryption 6) File Compression 7) Transfer. For each step the start and end times are recorded to obtain the total time spent by each group.The result of this test shown in fig. 2. The time spent on token generation, duplicate check, and Share Token is constant because they only use metadata. In contrast, other steps such as tagging, file compression, file encryption, and transfer, the time spent increases linearly with the file size because these operations involve in actual file data.

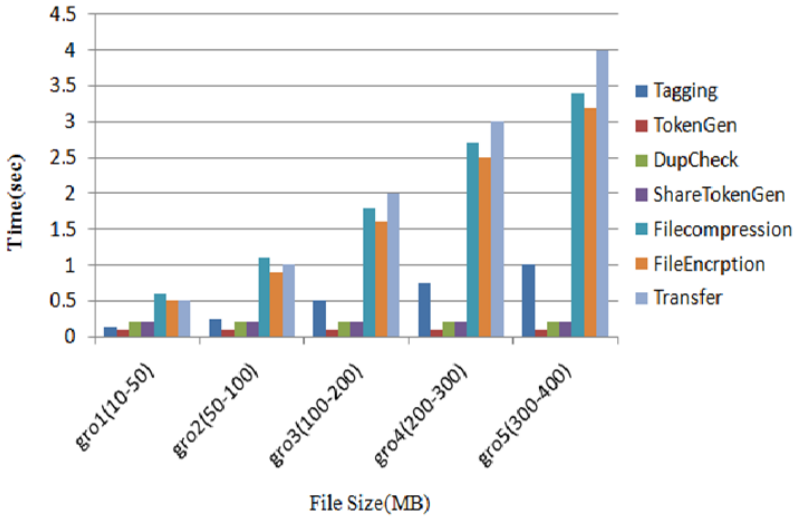

Figure 2. Time Breakdown for differently sized group files.

8.2 Effect of the duplication ratio:

To evaluate the duplication ratio effect in the system, two data sets each consists of 50 400MB files are prepared, then the first is uploaded set as an initial upload. The second data set is divided into six groups depend on the deduplication ratio, each group has the following ratio of duplicate (0-20-40-60-80-100)%. The average time of uploading the second set is shown in Fig. 3. The time spent on transfer, encryption, compression, duplication check decrease linearly with increasing the duplication ratio. Total time spent on uploading the file with deduplication ratio at 100% is only 10.44% of unique files.

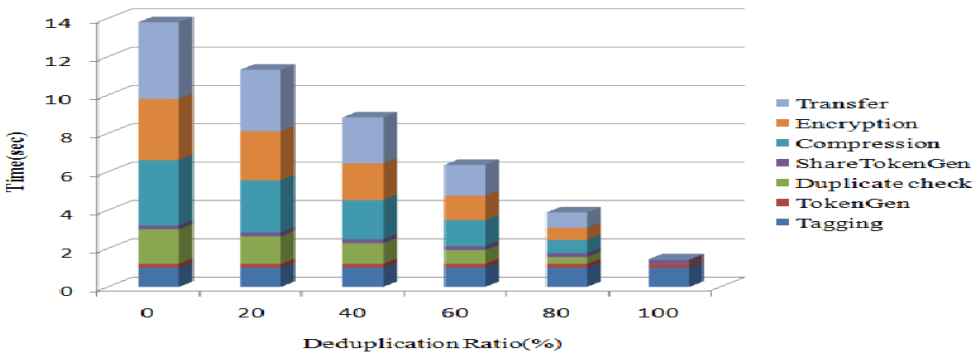

Figure 3 Time breakdown for deduplication ratio scenario

VIII. Conclusion and Future Work.

In this paper the concept of authorized deduplication was proposed to protect the data security and a novel deduplication mechanism is proposed employing both the public Cloud and private cloud by allowing them to play different roles in the process of deduplication. Deduplication technique has a lot of advantages, security and privacy of the sensitive data it's challenging by both inside and outside attackers. Proposed work provides space saving as well as security to the data, by use Token generation mechanism. In evaluation, the security analysis is performed against internal and external attackers, found that the system is mostly secure. A prototype is implemented for the proposed scheme and conducted test experiments on this prototype. It has been shown that the duplicate check scheme achieved minimal overhead. Currently proposed system designed for single cloud, in future, it may can extend to check duplicate files in multiple cloud servers accessed and shared by multiple users and also can be integrated with proofs of retrievability (POR) mechanisms [20][21] . There is also a possibility that multiple hash tables may be created for different types of files. This would increase the speed of lookup and is also collision free.

References

[1] Dropbox, a file-storage and sharing service. http://www.dropbox.com/.
[2] Google Drive. http://drive.google.com.
[3] MOZY. Mozy, a file-storage and sharing service. http://mozy.com/.
[4] Jin Li, Yan Kit Li, Xiaofeng Chen, Patrick P. C. Lee, Wenjing Lou. "A Hybrid Cloud Approach for Secure Authorized Deduplication ". In IEEE Transactions on Parallel and Distributed Systems, 2015.
[5] J. Yuan and S. Yu, "Secure and constant cost public cloud storage auditing with deduplication," in IEEE Conference on Communications and Network Security (CNS), 2013, pp. 145–153.
[6] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems," in Proceedings of the 18th ACM Conference on Computer and Communications Security. ACM, 2011, pp. 491–500.
[7] John Douceur, William Bolosky, and Marvin Theimer. US Patent 7266689: Encryption systems and methods for identifying and coalescing identical objects encrypted with different keys, 2007.
[8] Mihir Bellare, Sriram Keelveedhi, and Thomas Ristenpart. Dupless-Server aided encryption for deduplicated storage. In USENIX Security Symposium, 2013.
[9] M. Bellare, S. Keelveedhi, and T. Ristenpart. Message-locked encryption and secure deduplication. In EUROCRYPT, pages 296–312, 2013.
[10] 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 ICDCS, pages 617–624, 2002.
[11] Sven Bugiel, Stefan N¨urnberger, Ahmad-Reza Sadeghi, Thomas Schneider."Twin Clouds: Secure Cloud Computing with Low Latency". Center for Advanced Security Research Darmstadt, Technische Universit¨at Darmstadt, Germany.
[12] J. Yuan and S. Yu. Secure and constant cost public cloud storage auditing with deduplication. IACR Cryptology ePrint Archive, 2013:149, 2013.
[13] 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.
[14] J. Xu, E.-C. Chang, and J. Zhou. Weak leakage-resilient client-side de duplication of encrypted data in cloud storage. In ASIACCS, pages 195–206, 2013.
[15] 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.
[16] Zhang, Kehuan, et al. "Sedic: privacy-aware data intensive computing on hybrid clouds." Proceedings of the 18th ACM conference on Computer and communications security. ACM, 2011.
[17] 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.
[18] Z. Wilcox-O'Hearn and B. Warner. Tahoe: the least-authority filesystem. In Proc. of ACM StorageSS, 2008.
[19] S. Kamara and K. Lauter, ''Cryptographic Cloud Storage,'' in Proc. Financial Cryptography: Workshop Real-Life Cryptograph. Protocols Standardization, 2010, pp. 136-149.
[20] A. Giuseppe, B. Randal, C. Reza et al., "Provable data possession at untrusted stores," Proceedings of CCS, vol. 10, pp. 598–609, 2007.
[21] A. Juels and B. S. Kaliski, Jr., "Pors: proofs of retrievability for large files," in Proceedings of the 14th ACM conference on Computer and communications security, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 584–597