International Journal of Research in Engineering, Science and Management
Volume-2, Issue-3, March-2019
www.ijresm.com | ISSN (Online): 2581-5792


# Trust Worthy Deduplication of Encrypted Data in Cloud Storage

K. Kalpana Devi[1], G. Vasuki[2], B. Sowmya[3], P. Veeralakshmi[4]
[1,2]*Student, Department of Information Technology, Prince Shri Venkateshwara Padmavathy Engineering College, Chennai, India*
[3,4]*Professor, Department of Information Technology, Prince Shri Venkateshwara Padmavathy Engineering College, Chennai, India*

***Abstract*: At present, there is a networking technique. In storage services with huge data, the storage servers may want to reduce the volume of stored data, and the clients may want to monitor the integrity of their data with considerable increase in the amount of data stored in storage services, along with dramatic evolution of a low cost, since the cost of the functions related to data storage increase in proportion to the size of the data. To achieve these goals, secure deduplication and integrity auditing delegation techniques have been studied, which can reduce the volume of data stored in storage by eliminating duplicate copies and permit clients to efficiently verify the integrity of stored les by delegating costly operations to a trusted party, respectively. In this paper should be design a combined technique, which performs both secure deduplication of encrypted data and public integrity auditing of data. To support the two functions, the proposed scheme performs challenge-response protocols using the BLS signature-based Homomorphic linear proposed scheme satisfies all the fundamental security requirements. We also propose two variances that provide higher security and better performance has been proposed.**

***Keywords*: MD5 (Message Digest), AES (Advanced Encryption Standard)**

## 1. Introduction

In cloud storage services, clients outsource data to a remote storage and access the data whenever they need the data. Recently, owing to its convenience, cloud storage services have become widespread, and there

is an increase in the use of cloud storage services. Well-known cloud services such as Drop box and iCl are used by individuals and businesses for various notable change in information based services that has happened recently is the volume of data used in such services due to the dramatic evolution of network techniques. For example, in 5G networks, gigabits of data can be transmitted per second, which means that the size of data that is dealt by cloud storage services will increase due to the performance of the new networking technique. In this viewpoint, we can characterize the volume of data as a main feature of cloud storage services. Many service providers have already prepared high resolution contents for their service to utilize faster networks. For secure cloud services in the new era, it is important to prepare suitable security tools to support this change. Larger volumes of data require higher cost for managing the various aspects of data, since the size of data influences the cost for cloud storage services. The scale of storage should be increased according to the quantity of data to be stored. In this viewpoint, it is desirable for storage servers to reduce the volume of data, since they can increase their profit by reducing the cost for maintaining storage. On the other hand, clients are mainly interested in the integrity data stored in the storage maintained by service providers. To verify the integrity of stored files, clients need to perform costly operations, whose complexity increases in proportion to the size of data. In this viewpoint, clients may want to verify the integrity with a low cost regardless of the size of data. Owing to the demands of storage servers and clients, many researches on this topic are available in the literature. To reduce the volume of data, deduplication has to be performed in servers so that the storage space efficiency can be improved by removing duplicated copies. According to the research report of EMC, about 75% of the data are duplicated [7]. This fact raises the need for design of deduplication technology. In the literature, there are studies on two types of deduplication techniques. Among them, client-side deduplication has attracted the interest of researchers more than server-side deduplication due to its efficiency in computation and communication. Unfortunately, client-side deduplication has a number of problems. When clients use cloud storage services, the integrity of stored data is the most important requirement. In other words, clients want to be guaranteed about the integrity of their data in the cloud. In cloud storage services, we cannot exclude the possibility of weak cloud servers, which are vulnerable to internal and external security threats. In the case of data loss due to some incident, weak servers may try to hide the fact that they lost some data, which were entrusted by their clients. More seriously, servers delete rarely accessed users' data in order to increase the profit. Therefore, it is a natural requirement of clients to periodically check the current state of their data. To do this in practice, we need a way to efficiently check the integrity of data in remote storage Secure deduplication and integrity auditing are fundamental functions required in cloud storage services. Hence, individual researches have been actively conducted on

these two topics. However, relatively few studies have been conducted for designing a combined scheme that can support these two functions at the same time. The fundamental goal of the design of a combined model is to guarantee less overhead than a trivial combination of existing schemes. In particular, the goal of this paper is to improve the cost of both computation and communication. In this paper, we design a new scheme for secure and efficient cloud storage service. The scheme supports both secure deduplication and integrity auditing in a cloud environment. In particular, the proposed scheme provides secure deduplication of encrypted data. Our scheme performs PoW for secure deduplication and integrity auditing based on the homomorphic linear authenticator (HLA), which is designed using BLS signature. The proposed scheme also supports public auditing signature TPA (Third Party Auditor) to help low-powered clients. The proposed scheme states all fundamental security requirements, and is more efficient than the existing schemes that are designed to support deduplication and public auditing at the same time. Note that the preliminary version of this paper appeared in Moby Sec2017[16]. The main improvement in this paper is that we propose two variations to provide higher security and better performance. In the first variance, which is designed for stronger security, we assume a stronger adversary and provide a countermeasure against the adversary. In the second variance, we design a technique that supports a very low-powered client and entrusts more computation to the cloud storage server in the upload procedure. This paper is organized as follows. Section II describes related works. In Section III, we propose a secure deduplication technique, which supports integrity auditing based on AES signature, and analyze it in Section IV. In addition, we suggest two improved protocols in Section V. Finally, Section VI provides the conclusion.

## 2. Related works

Secure deduplication is interesting for both industrial and research communities; therefore, several secure deduplication schemes have been proposed showed some attacks in the case of client-side deduplication, which causes data leakage. To counter the attacks, the concept of Paw was introduced in Later, in the convergent encryption, which is defined as message-locked encryption, was formalized and then, Bellaire et al. presented another scheme called Duplets for semantic security. To support data integrity, two concepts, PDP and POR, have been introduced PDP for ensuring that the cloud storage providers actually possess the ales without retrieving or downloading the entire data. It is basically a challenge-response protocol between the verifier (a client or TPA) and the prover (a cloud). Compared to PDP, POR not only ensures that the cloud servers possess the target files, but also guarantees their full recovery. Since then, a number of POR schemes and PDP schemes have been proposed. A simple combination of two independent techniques designed for the two above mentioned issues does not efficiently deal with the issues at once, because achieving storage efficiency contradicts with the deduplication of authentication tags. In public auditing with a deduplication scheme based on homomorphic linear authentication tags was proposed. Each user has to generate the integrity tags, even for the file in the cloud. Moreover, the file is available in its plain form on the cloud side. The proposed an integrity auditing scheme for encrypted deduplication storage. This scheme is based on homomorphic verifiable tags and Merkle hash tree. A user encrypts his file by using a convergent encryption technique and uploads the file to a fully trusted TPA.

To reduce the volume of data, deduplication has to be performed in servers so that the storage space efficiency can be improved by removing duplicated copies. According to the research report of EMC, about 75% of the data are duplicated. In the literature, there are studies on two types of deduplication techniques. Among them, client-side deduplication has attracted the interest of researchers more than server-side deduplication due to its efficiency in computation and communication. Unfortunately, client-side deduplication has a number of problems. When clients use cloud storage services, the integrity of stored data is the most important requirement. In other words, clients want to be guaranteed about the integrity of their data in the cloud. In cloud storage services, we cannot exclude the possibility of weak cloud servers, which are vulnerable to internal and external security threats. In the case of data loss due to some incident, weak servers may try to hide the fact that they lost some data, which were entrusted by their clients. More seriously, servers delete rarely accessed users' data in order to increase the port.

## 3. The proposed scheme

Here, we describe the system model of our scheme. We also give the corresponding security model. After that, we will give a detailed description of our scheme according to the models. In this paper, we design a new scheme for secure and efficient cloud storage service. The scheme supports both secure deduplication and integrity auditing in a cloud environment. In particular, the proposed scheme provides secure deduplication of encrypted data. Our scheme performs MD5 hash function for secure deduplication and integrity auditing. The proposed scheme also supports public auditing using a TPA (Third Party Auditor) to help low-powered clients. The proposed scheme satisfies all fundamental security requirements, and is more efficient than the existing schemes that are designed to support deduplication and public auditing at the same time. The main improvement in this paper is that we propose two variations to provide higher security and better performance. In the first variance, which is designed for stronger security, we assume a stronger adversary and provide a countermeasure against the adversary. In the second variance, we design a technique that supports a very low-powered client and entrusts more computation to the cloud storage server in the upload procedure.

*A. System and security model*

Our scheme utilizes the BLS signature-based Homomorphic Linear Authenticator (HLA), which was proposed in [14], for integrity auditing and secure deduplication. We also introduce TPA to support public integrity auditing. The proposed scheme consists of the following entities.

- Client (or user): Outsources data to a cloud storage. CE-encrypted data is rst generated, and then uploaded to the cloud storage to protect confidentiality. The client also needs to verify the integrity of the outsourced data. To do this, the client delegates integrity auditing to the TPA.
- Cloud Service Provider (CSP): Provides data storage services to users. Deduplication technology is applied to save storage space and cost. CSP has significant resources to govern distributed cloud storage servers and to manage its database servers. It also provides virtual infrastructure to host application services. These services can be used by the client to manage his data stored in the cloud servers.
- TPA (Third Party Auditor): Performs integrity auditing on behalf of the client to reduce the client's processing cost. Instead of the client, the auditor sends a challenge to the storage server to periodically perform an integrity audit protocol. TPA is assumed to be a semi-trust model, that is, an honest but curious model. Under the assumption, it is assumed that the TPA does not collude with other entities. The relation between entities can be seen in Fig. 1. A client and a CSP perform PoW for secure deduplication, and a TPA is placed between the client and the CSP to execute integrity auditing instead of the client. Here, we consider the following types of adversary models: outside adversary, insider adversary CSP, and semi-honest adversary TPA.

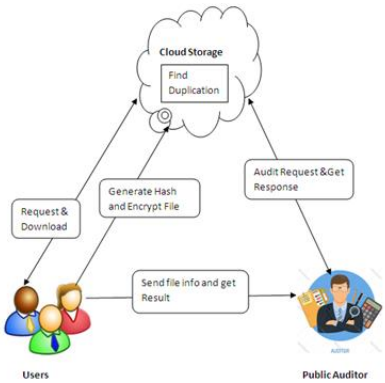



Fig. 1. System model

*Outside adversary:* Assuming that the communication channel is not secure, an outside attacker can easily intercept the transmitted data. An outside attacker attempts to pass the PoW process as if it were the proper owner of the data. The CSP assumes that it can act maliciously. It attempts to get information out of the user's encrypted data, and modify or delete the user's data. The TPA is assumed to perform the protocol correctly; however, in the process it tries to obtain information about the user's data. In addition, the proposed scheme should satisfy the following security objectives. Except for the information about duplication, no information about the outsourced data is revealed to an adversarial party. Secure Deduplication is supported without revealing any information except for the information about duplication. The TPA is able to examine the accuracy and availability of the outsourced data without querying the entire data and without intervention by the data owner. If the CSP is keeping the user's data intact, it can pass the TPA's verification.

*B. Detailed operation of proposed method*

- *Register Login:* Here each user has to register by giving their own information to become a cloud user and create an authentication to use the cloud service provider (CSP) and third party authority (TPA) also have the authentication to login to the cloud. Upload Files and find
- *Deduplication:* After successful authentication, user can check their data in the cloud. If the user wants to upload a file then they can browse the file and upload it. Before uploading a file is first checked by user whether it is already found or not by generating the hash values of the file and compare it with own files. If already exist, then file will not save in cloud else file will be encrypted and saved in cloud. User sends the file information to the TPA for auditing the files regularly.
- *Audit Request and Response:* In this model, TPA sends the audit request to the cloud by selecting the files randomly and waiting the response from the cloud. After receiving the request from the TPA the cloud generate byte values of a file and send it to the TPA for verification. TPA verifies a byte values from the cloud with customer given byte values to check the file is safe or not. TPA sent the audited result to the corresponding user to check the status of a file.
- *Download Request and Response:* To download a file from the cloud, User has to send the request to cloud. After receiving the request from the user the CSP verifies the cloud user information and accept the request. Accepted request is shown to the user to download a file from the cloud. After downloading a file, user decrypt a file and get the original file.

*C. Working procedure*

It compares objects (usually files or blocks) and removes objects (copies) that already exist in the data set. The deduplication process removes blocks that are not unique.

- Divide the input data into blocks or “chunks.”
- Calculate a hash value for each block of data.

- Use these values to determine if another block of the same data has already been stored.

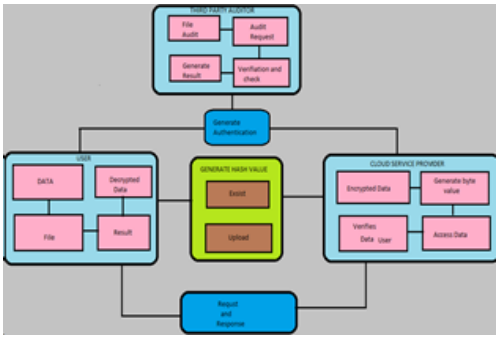


Fig. 2. Working procedure

### *D. Data Deduplication*

Once the data is chunked, an index can be created from the results, and the duplicates can be found and eliminated. Only single instance of data is stored The actual process of data deduplication can be implemented in a number of different ways. We can eliminate duplicate data by simply comparing two files and making the decision to delete one that is older or no longer needed. File system-based deduplication is a simple method to reduce duplicate data at the file level. An example of this method is comparing the name, size, type and date-modified information of two files with the same name being stored in a system. If these parameters match, you can be pretty sure that the files are copies of each other and that you can delete one of them with no problems. Although this example isn't a foolproof method of proper data deduplication, it can be done with any operating system and can be scripted to automate the process, and best of all, it's free. Based on a typical enterprise environment running the usual applications, you could probably squeeze out between 10 percent to 20percent better storage utilization by just getting rid of duplicate files.

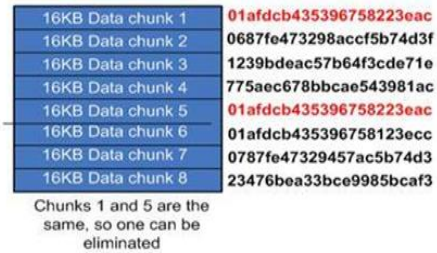



Fig. 3. Deleting duplicate files

### *E. AES algorithm*

Broadly speaking the encryption/decryption can be done via symmetric key or asymmetric key. In symmetric algorithms, both parties share the secret key for both encryption/decryption, and from privacy perceptive it is important that this key is not compromised, because cascading data will then be compromised. Symmetric Encryption/decryption require less power for computation. On the other hand, asymmetric algorithms use pairs of keys, of which one key is used for encryption while other key is used for decryption. Generally, the private key is kept secret and generally held with the owner of data or trusted 3rd party for the data, while the public key can be distributed to others for encryption. The secret key can't be obtained from the public key. In our case since the encryption/decryption is performed on trusted 3rd party server, symmetric key is used, and it delegates the burden of key management to the trusted 3rd party. If key management where to be done at clients end it would mean, 1. either they have to remember the big key 2. store the key in all devices/machine which will be used to access the cloud services, which make user device a bottleneck. 3. individual owner has to take the responsibility of sharing the key with specific authorized group of user which he/she dene. Outline of the AES Algorithm Constants: intNb = 4; // but it might change someday int Nr = 10, 12, or 14; // rounds, for Nk = 4, 6, or 8 Inputs: array in of 4*Nb bytes // input plaintext array out of 4*Nb bytes // output ciphertext array w of 4*Nb*(Nr+1) bytes // expanded key Internal work array: state, 2-dim array of 4*Nb bytes, 4 rows and Nb cols Algorithm: void Cipher(byte[ ] in, byte[ ] out, byte[ ] w) f byte[ ][ ] state = new byte[4][Nb]; state = in; // actual component-wise copy AddRoundKey(state, w, 0, Nb - 1); // see Section 4 below for (int round = 1; round < Nr; round++) f SubBytes(state); // see Section 3 below ShiftRows(state); // see Section 5 below MixColumns(state); // see Section 5 below AddRoundKey(state, w, round*Nb, (round+1)*Nb - 1); // Section 4 gSubBytes(state);//see Section 3 below ShiftRows(state); // see Section 5 below 20 AddRoundKey(state, w, Nr*Nb, (Nr+1)*Nb - 1); // Section 4 out = state;

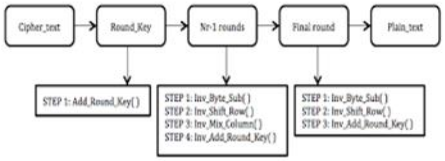



Fig. 4. Sequence of AES Algorithm

### *F. MD5 algorithm*

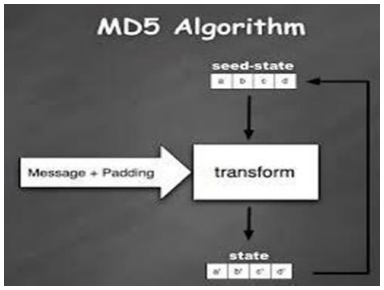



Fig. 5. MD5 algorithm

It generates 16-bit hash key to the file positioned for encryption. At the same time user generates a master key for the file to have authenticated access by which file can be downloaded and decrypted. The successful encrypted file is

uploaded to the auditor. Insegment1 this job is carried out. Coming to redundant duplication, hash code generated by MD5 in background for every encryption is sorted and maintained by the auditor. Here the specialty of MD5 is generating hash code for the file irrespective of names. This hash code helps the manager to cross check hashes for each and every time. This method also uses to store some hash code even then it doesn't lead to over heading on manager auditor end, this is because the hash code generated by MD5 algorithm is 16 characters key, are same for even 1000 characters file. This hash code even differs even if one character in the file differ. So MD5 is a highly strengthened idea for redundant duplication. There by this hash stored in auditor for each file will decide whether to traverse the file or not to the cloud. Here by it reduces and replaces the traditional idea of comparing the file names for the existence in cloud. Here vital role is played by the auditor to compare and cross check content of the file data. In regular interval of time the auditor often logs in to have a check for replication to avoid duplicates. According to auditor observation if the same content of the files is keep on outsourced then it leads to wastage of cloud space which cause impact on over heading and performance of cloud and drain the efficiency of the cloud. All the current hash code of the file is cross check with the existing code. Memory constraints will not be affected by the key stored. Memory consumed by just 128bit will not occupy heavy space in the audit dynamo, Therefore MD5 hashing technique gives very less overheating which is not a very big deal. Here hash code comparison is not at all carried out by the client end nor does cloud server, it is done separate by the third party server called auditor. No the client server or cloud server is responsible for the generation of the hash code even for storing and comparing. Consequently, this method is more accurate and reliable way of auditing and monitoring the redundant duplicates, nonredundant in cloud. As cloud cost for storing the data with repetition of file leads to wastage of money to the client. If file is very small then repetition is considerable, if it is in terabytes or terabytes repetitions is not suggestible.

- Public Key Generation For each and every current input of client a random unique key is generated known as public key. This key is generated for every login and key is very much essential for access of cloud. Next client uploads the file f1 and call for encryption to encrypt, let the encrypted file be f2, clientele ways want to secure the data so uploads the encrypted file. Input (f1, f2) Upload ( ) In this process cypher texted file taken as input and ask the client for desired name to upload the file, The name of the file is displayed in the list of file
- Hash Key Generation Here MD5 is used which runs on background of auditor. This MD5 generates 16 char hash key Auditor will perform all the process of finding duplications and store the key in the log for future checking.
- Redundant Duplication Here comparing of hash key for the new hash.

INPUT (F1, F2) DES (F1, F2) =£1,£2; MD5 (F1, F2) =f1#,f2#; UPLOAD ((£1, f1#) ∩ (£2, f2#));

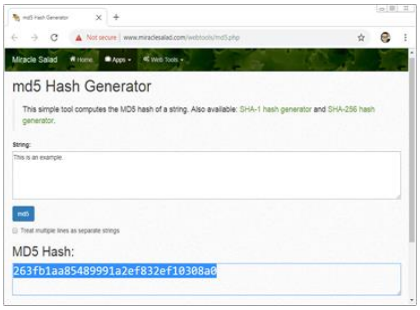



Fig. 6. Md5 hash generator

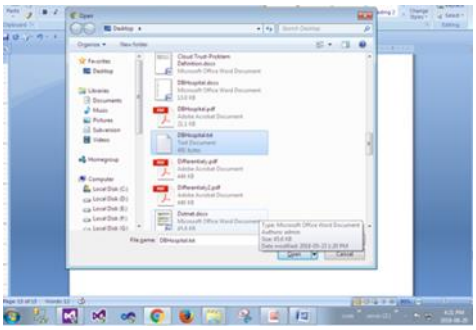
*1) Input*

MD5 hashes are 128-bits in length and are normally shown in their 32 digit hexa decimal value equivalent. This is true no matter how large or small the file or text may be.
Here's an example:

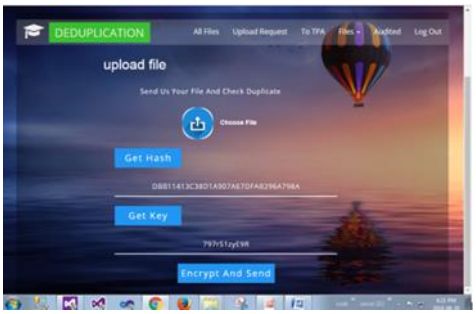
- Plain text: This is a test.
- Hex value: 120EA8A25E5D487BF68B5F7096440019

When more text is added, the hash translates to a totally different value but with the same number of characters:
Plain text: This is a test to show how the length of the text does not matter.
Hex value: 6c16fcac44da359e1c3d81f19181735b.

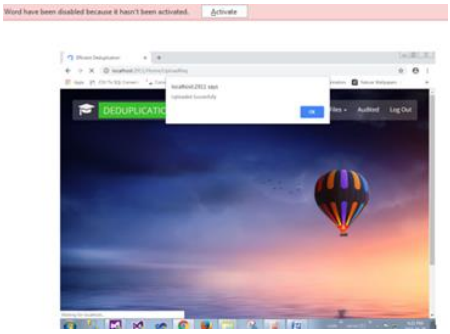
*2) Screenshots*



(a)




(b)



(c)
Fig. 7. File upload process

## 4. Conclusion

We proposed a scheme to achieve both secure deduplication and integrity auditing in cloud environment. To prevent leakage of important information about user data, the proposed scheme supports a client side deduplication of encrypted data, while simultaneously supporting public auditing of encrypted data. We used MD5 and AES algorithm to compute authentication tags for the integrity auditing. The proposed scheme satisfied the security objectives, and improved the problems of the existing schemes.

## 5. Future enhancement

To support the dynamic auditing, we will develop a dynamic provable data possession protocol based on cryptographic hash function and symmetric key encryption. Their idea is to pre-compute a certain number of metadata during the setup period, so that the number of updates and challenges is limited and fixed beforehand. In their protocol, each update operation requires recreating all the remaining metadata, which is problematic for large files. Moreover, their protocol cannot perform block insertions anywhere (only append-type insertions are allowed).

## References

[1] G.Atenieseet al., “Provable data possession at untrusted 0 stores," in Proc. 14th ACM Conf. Comput. Commun. Secur., Alexandria, VA, USA, 2007, pp. 598-609.

[2] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, ``Scalable and ef_cient provable data possession," in Proc. 4th Int. Conf. Secur. Privacy Commun. Networks, Istanbul, Turkey, 2008, Art. no. 9.

[3] D. Boneh, B. Lynn, and H. Shacham, ``Short signatures from the Weil pairing," J. Cryptol., vol. 17, no. 4, pp. 297-319, 2004.

[4] Y. Dodis, S. Vadhan, and D. Wichs, ``Proofs of retrievability via hardness ampli_cation," in Proc. 6th Theory Cryptogr. Conf., San Francisco, CA, USA, 2009, pp. 109-127.

[5] M. Dworkin, ``Recommendation for block cipher modes of operation: Methods and techniques," NIST, Gaithersburg, MD, USA, Tech. Rep. SP-800-38A, 2001.

[6] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, ``Dynamic provable data possession," in Proc. 16th ACM Conf. Comput. Commun. Secur.Chicago, IL, USA, 2009, pp. 213-222.

[7] J. Gantz and D. Reinsel, ``The digital universe decade_Are you ready?" International Data Corporation, China Oceanwide, MA, USA, White Paper IDC-925, 2010.

[8] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, ``Proofs of ownership in remote storage systems," in Proc. 18th ACM Conf. Comput. Commun. Secur. Chicago, IL, USA, 2011, pp. 491-500.

[9] D. Harnik, B. Pinkas, and A. Shulman-Peleg, ``Side channels in cloud services: Deduplication in cloud storage," IEEE Security Privacy, vol. 8, no. 6, pp. 40_47, Nov./Dec. 2010.

[10] A. Juels and B. S. Kaliski, Jr., ``Pors: Proofs of retrievability for large _les," in Proc. 14th ACM Conf. Comput. Commun. Secur., Alexandria, VA, USA, 2007, pp. 584_597.

[11] S. Keelveedhi, M. Bellare, and T. Ristenpart, ``DupLESS: Server-aided encryption for deduplicated storage," in Proc. 22nd USENIX Security. Symp., Washington, DC, USA, 2013, pp. 179-194.

[12] J. Li, J. Li, D. Xie, and Z. Cai, ``Secure auditing and deduplicating data in cloud," IEEE Trans. Comput., vol. 65, no. 8, pp. 2386-2396, Aug. 2016.

[13] X. Liu, W. Sun, H. Quan, W. Lou, Y. Zhang, and H. Li, ``Publicly veri_able inner product evaluation over outsourced data streams under multiple keys," IEEE Trans. Services Comput., vol. 10, no. 5, pp. 826_838, Sep./Oct. 2017.

[14] H. Shacham and B. Waters, ``Compact proofs of retrievability," in Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur., Melbourne, VIC, Australia,2008, pp. 90-107.

[15] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, ``Enabling public auditability and data dynamics for storage security in cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 5, pp. 847-859, May 2011.

[16] T. Y. Young, K. Y. Chang, K. R. Rhee, and S. U. Shin, ``Public audit and secure deduplication in cloud storage using BLS signature," Res. Briefs Inf. Commun. Technol. Evol., vol. 3, pp. 1_10, Nov. 2017, Art. no. 14.

[17] J. Yuan and S. Yu, ``Proofs of retrievability with public veri_ability and constant communication cost in cloud," in Proc. Int. Workshop Secur. Cloud Comput. Hangzhou, China, 2013, pp. 19-26.

[18] J. Yuan and S. Yu, ``Secure and constant cost public cloud storage auditing with deduplication," in Proc. IEEE Conf. Commun. Netw. Secur. (CNS)National Harbor, MD, USA, Oct. 2013, pp. 145-153.