J Inf Process Syst, Vol.17, No.3, pp.658~673, June 2021
https://doi.org/10.3745/JIPS.03.0160

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Survey on Data Deduplication in Cloud Storage Environments

Won-Bin Kim* and Im-Yeong Lee*

Abstract

Data deduplication technology improves data storage efficiency while storing and managing large amounts of data. It reduces storage requirements by determining whether replicated data is being added to storage and omitting these uploads. Data deduplication technologies require data confidentiality and integrity when applied to cloud storage environments, and they require a variety of security measures, such as encryption. However, because the source data cannot be transformed, common encryption techniques generally cannot be applied at the same time as data deduplication. Various studies have been conducted to solve this problem. This white paper describes the basic environment for data deduplication technology. It also analyzes and compares multiple proposed technologies to address security threats.

Keywords

Date Deduplication, Cloud Storage, Encryption, Security

1. Introduction

The rapid development of information and communication technology (ICT) has induced various changes to data storage environments. In the early computing environment, punch cards, magnetic tapes, etc., were used as auxiliary memory devices. Since then, auxiliary storage has made considerable progress in terms of speed and data integration, from hard disk drives to flash memory. Consequently, it was predicted that the lack of storage space would be solved by the increase in data density. However, improved data processing technology developed have improving the productivity and quality of the data, thereby increasing its volume. Therefore, it is still important to secure the storage space available in a computing environment.

Today, the primarily used auxiliary devices include hard disk drives, solid-state drives, and flash memory. The advantages of such storage mediums are that they can store a large amount of data and are portable. However, they are always exposed to loss and failure, and the stored data can only be accessed by carrying the storage medium around. Therefore, to solve this problem, storage mediums have become lighter, more compact, and recoverable even in case of failure. However, carrying storage mediums around is still inconvenient. Cloud storage has emerged as a good alternative to this problem.

Cloud storage is a type of cloud service that can be viewed as a form of remote storage available over

Manuscript received February 1, 2019; first revision July 10, 2019; Second revision September 25, 2019; accepted October 6, 2019.
Corresponding Author: Im-Yeong Lee (imylee@sch.ac.kr)
* Dept. of Software Convergence, Soonchunhyang University, Asan, Korea (wbkim29@sch.ac.kr, imylee@sch.ac.kr)

a network. When connected to the network, the service is freely received without any restrictions on place, space, and time, thus addressing the problems of carrying physical storage mediums and their vulnerability to loss and failure. In addition, using cloud storage through a cloud service provider allows us to add or reduce storage space as needed, providing flexibility to the storage space according to the amount of data.

Cloud storage provides remote storage over a network, allowing multiple users to access and use it at the same time. Therefore, you should always reserve available storage space because of the additions of various data. Data deduplication technology is designed to avoid storing the same data repeatedly to save storage space [1-12].

Data deduplication technology is a technology designed to efficiently store data in a storage space by preventing us from repeatedly storing the same data in the same storage. However, each computing environment has different requirements, and different technologies must be applied to meet them. A cloud storage environment is an environment where storage is available remotely over a network. In general, remote servers are treated as semi-trusted environments, so you need to take appropriate action to account for data leakage from the insider or from an outside party [13]. Therefore, data stored in cloud storage must be encrypted, and contain no information that can be used to recover the source data. Section 2 describes the types of data deduplication techniques, Section 3 describes security issues in data deduplication, Section 4 describes security technologies used for secure data deduplication and Section 5 describes the trends in deduplication systems that use these technologies.

2. Data Deduplication

This section examines the various forms and application technologies that can be used to perform deduplication in cloud storage.

2.1 Classification of Deduplication Locations

Data deduplication for cloud storage can be categorized as client-side deduplication, appliance deduplication, or server-side deduplication, depending on where the data is deduplicated [14], as shown in Fig. 1.

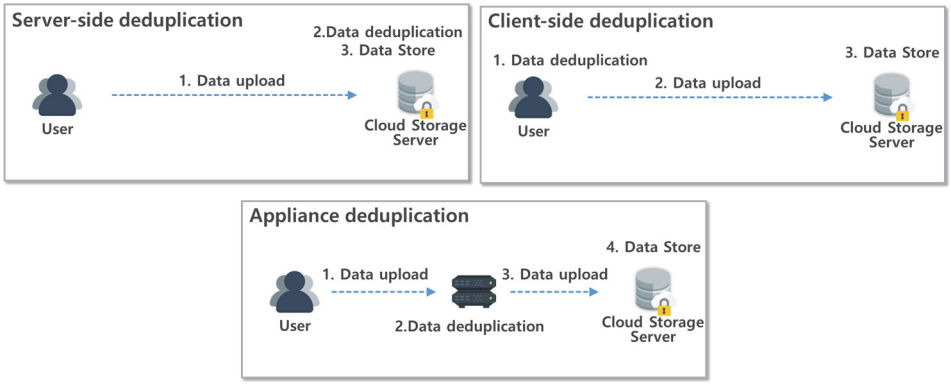

Fig. 1. Classification of deduplication locations.

2.1.1 Server-side deduplication

Receiving the source of the file and handling deduplication in cloud storage is called server-side deduplication. This method is distinctive in that data deduplication processing can be performed without imposing an additional burden on the user. Furthermore, the hardware performance of a cloud storage service is much better than that of general user devices. This results in a faster processing speed than deduplication on the user's device. However, because all the source data that is not deduplicated is transmitted, if the communication environment is poor, the user may face a heavy burden while uploading the data [1,14].

2.1.2 Client-side deduplication

If data is deduplicated at the user's terminal and only the deduplicated data is transmitted to the server, the user will generate low communication traffic as the redundancy ratio of data increases. Client-side deduplication technology is designed for this purpose. The client-side deduplication method checks the list of data stored in cloud storage on the user's device and sends only unsaved data. Therefore, the higher the overlap ratio of the data to be uploaded and the data stored in the server, the lesser the data to be uploaded, so even if the communication environment is poor, duplicate data can be easily deleted. However, as described above, since a general user's device has a lower processing speed than the hardware of the cloud storage service, the user's device may be burdened by the process of deduplication [14].

2.1.3 Appliance deduplication

Appliance deduplication is a combination of server-side deduplication and client-side deduplication. This approach places an appliance between the client and the server, which receives the uploaded data instead of the server. Subsequently, the appliance performs deduplication and sends only non-redundant data to the server. The advantages of this are that it is unnecessary to perform deduplication on the user's device and the load on the server and client side is reduced. However, if the number of users increases, the appliance may become a bottleneck. Therefore, it is necessary to install additional appliances according to the number of users, but this is directly linked to an increase in cost [13].

2.2 Classification of Deduplication Levels

The data deduplication environment is classified as file-level deduplication or block-level deduplication according to the deduplication level. The block-level deduplication method is further classified as either a fixed-length method, or a variable-length method. The detailed form [14] is explained below.

2.2.1 File-level deduplication

The initial data deduplication technique used a file deduplication method that compared data stored in the data storage to the data to be uploaded as a whole. File-level deduplication is also referred to as single-instance storage (SIS) and is used for storage systems that require fast discovery. This method involves hashing the file itself to generate abbreviated hash data, and then comparing the abbreviated hash data to the hash data of the files stored on the storage server. Therefore, the data source is stored in its source form. If some of the files stored in the data storage are modified and restored, both the source and modified files are kept intact, making it easy to manage versions of frequently modified files. In addition, since it is a method for comparing the files themselves, it is possible to search and compare very quickly, so

the data deduplication speed is faster than it is for the block-level deduplication method. However, due to the characteristics of the hash algorithm, the entire hash value changes even if a single bit of the source data is changed. Therefore, if the source data stored in the storage server is different by even a single bit, the corresponding data is recognized as different data and is not deduplicated [14] as shown in Fig. 2.

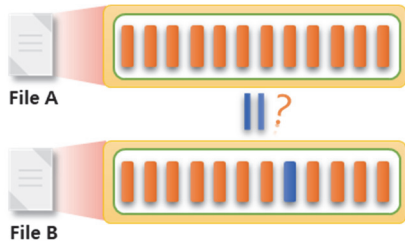

Fig. 2. File-level deduplication.

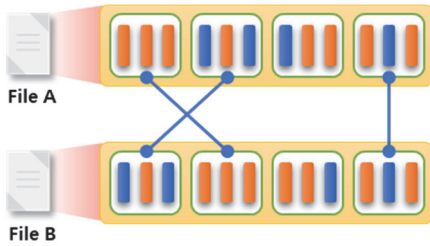

Fig. 3. Block-level deduplication.

2.2.2 Block-level deduplication

In the block-level deduplication method, the source file is divided into chunks (or segments) of block levels, and deduplication is performed. In block-level deduplication, if the data in the source file changes partially, only the unchanged blocks are deduplicated because these are identical to the corresponding blocks in the source file, as shown in Fig. 3. Therefore, deduplication efficiency is higher than in file-level deduplication methods, where deduplication is not performed even if only a single bit of the file is different. On the contrary, as the source file is divided into blocks, and as hashing and data comparison for deduplication is performed for each of them, the processing speed is slower than that of the file-level deduplication method. Therefore, the block-level deduplication method is mainly used for backup data storage that stores large amounts of data and does not require a fast search speed [14].

Fixed-length block-level deduplication

The block-level deduplication scheme is divided into a fixed-length scheme and a variable-length scheme according to the length of the block. Both methods are the same in that the source file is divided into blocks and deduplicated. However, in the fixed-length method, as the blocks are sequentially formed starting from the first data bit of the file, the method is inefficient when some data bits are added to or removed from the middle of a file, as shown in Fig. 4. Therefore, a variable-length method has been studied to solve the problem of the deduplication efficiency degradation of the fixed-length method [14].

Variable-length block-level deduplication

The variable-length method groups blocks consisting of a variable number of data bits rather than a fixed number, as is the case with the fixed-length method. Using this method, even if the data bits are

added to or removed from the middle of the source file, these bits are skipped, and blocks are formed again from the subsequent data bits to enable deduplication as shown in Fig. 5. The variable-length method is designed to maximize deduplication efficiency, while the file-by-file deduplication efficiency is significantly higher than that of the block deduplication method. However, as each block is constructed with n individual blocks, $n + 1$ individual blocks, and $n - 1$ individual blocks, it is a relatively slow process to generate a comparison of all cases by using algorithms such as Rabin Hash. As a result, the variable-length block deduplication approach is not used for the secure data deduplication described later, because encrypting blocks of variable lengths will waste both the computation power and the data storage space [14] (Table 1).

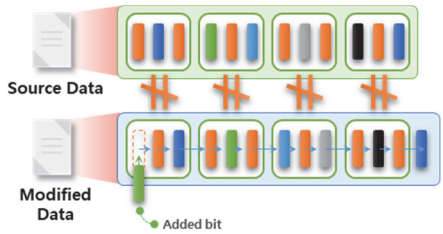

Fig. 4. Fixed-length block-level deduplication.

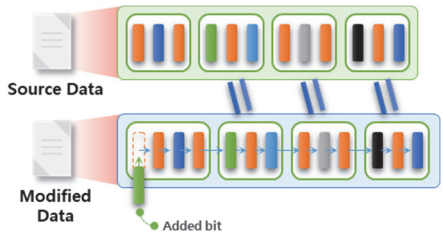

Fig. 5. Variable-length block-level deduplication.

Table 1. Comparison of data deduplication classifications

	Classification	Computation volume	Communication volume	Traffic volume	Storage space efficiency
Location	Server-side	High	Low	High	-
	Client-side	Low–High	High	Low	-
	Appliance	High (appliance)	High (appliance)	High (user)	-
		Low (user, server)	Low (user, server)	Low (appliance, server)	-
Level	File-level	Low	-	-	Low
	Block-level				
	Fixed-length	High	-	-	High
	Variable-length	Very high	-	-	Very high

3. Secure Data Deduplication

In a cloud storage environment, data deduplication technologies create various security issues in the process of storing data on a remote server [1-14]. This chapter summarizes these security issues.

3.1 Necessity for Secure Data Deduplication

A common deduplication technique is technology without encryption. If the data stored in the data storage is not encrypted, data cannot remain confidential if the data is leaked by internal or external attacks. Therefore, encryption is applied to ensure confidentiality. Encryption technologies make it impossible to know the contents of two encrypted file until decryption is performed, because different ciphers are generated when different users encrypt the same data with their own encryption keys. Therefore, because deduplication determines redundancy through the comparison of data, it is impossible to perform deduplication on encrypted data. This means that a special encryption technique must be applied for the deduplication of encrypted data. To do this, convergent encryption (CE), a technique for encrypting a data source by generating an encryption key, has been proposed [1-49].

3.2 Dictionary Attack

In the encrypted data-deduplication technique, the key-generation method generally uses CE technology. With this technology, even if different users encrypt the same source data, the same ciphertext is generated and can be deduplicated. However, since the same encryption key is always generated by the same source data, the encryption key and the encryption data can be generated by arbitrarily generating data that is not actually owned. An example is an in-house document. If an in-house document has a specific format and the contents of the form are limited, an attacker can list data that can be included in the file of the form and generate the data, key, and encrypted data by trying them one-by-one. If you can specify any type of data that can fit in the form, then a dictionary-attack threat can occur. Therefore, to solve these problems, it is necessary to improve the weakness of the existing CE method.

3.3 Poison Attack

In the client-side deduplication method, the user transmits a list of the data to be uploaded to the server, and the server transmits a list of the data that is not duplicate to the user. The user transmits only the non-duplicate data to the server based on the received list. However, in this process, when the user uploads the required data differently from the actual upload data, the information regarding the data stored in the server and the actual data become inconsistent. In this case, user 1 who uploads data A for the first time is actually an attacker who uploads data A′, and user 2 who subsequently uploads data A does not actually

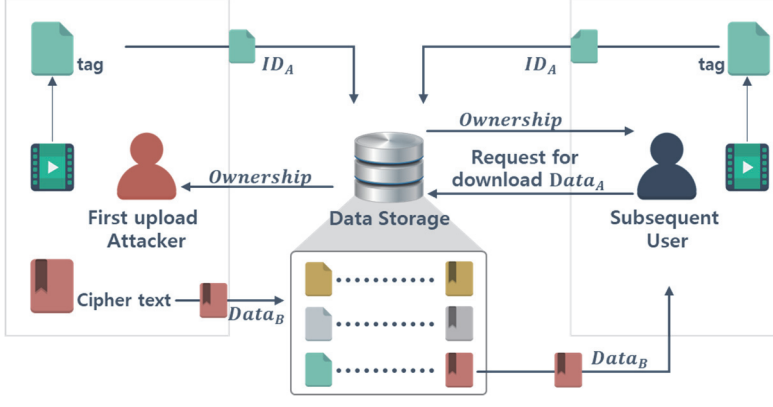

Fig. 6. Outline of poison attack.

upload data A through deduplication. Thereafter, if user 2 requests data A from the server, it obtains A′ instead of A. These security threats are poison attacks, and if data A′ contains malicious code, malicious code threats will occur (Fig. 6).

3.4 Ownership Forgery Attack

In client-side deduplication, the data to be uploaded by the user is hashed into an identifier, and the duplicate is checked using this identifier. In this case, if the data is already stored in the cloud storage, after confirming duplication, the user is given ownership of the data. Therefore, if the identifier of the data can be modified or modified, the ownership of the data stored in the cloud storage can be acquired without the data source [16,17] (Fig. 7).

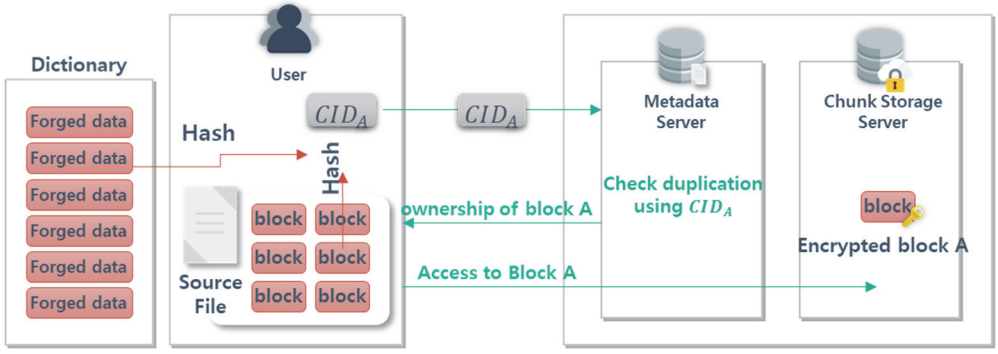

Fig. 7. Outline of ownership forgery attack.

4. Security Technology for Secure Data Deduplication

4.1 Convergent Encryption

CE is a cryptographic scheme proposed by Douceur et al. [2] in 2002. This technique hashes the data source to generate an encryption key and encrypts the data using this key. This method allows duplicate data to be removed because the same ciphertext is always generated even if different users encrypt the same source data. Most secure data deduplication techniques perform encryption based on CE, but the source data used to generate keys for CEs can be predicted, which can lead to threats such as dictionary attacks (Fig. 8). A method to solve this problem has been proposed since then [19-21].

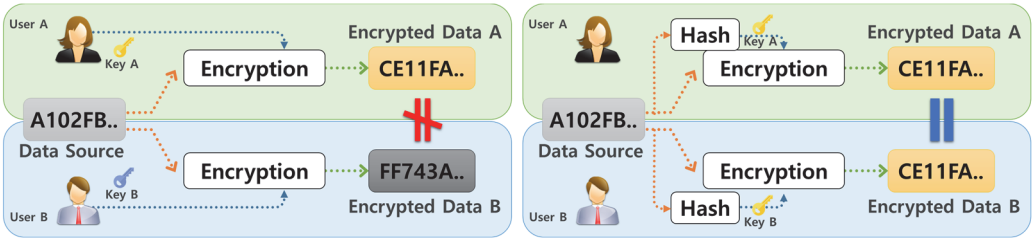

Fig. 8. General encryption method (top) and convergent encryption (bottom).

4.2 Prepare for Dictionary Attacks

In data deduplication technologies, CE is mainly used for data encryption. However, since CE uses the encryption key derived from the data source, a dictionary attack may occur if the source can be deduced. Therefore, this section describes how to defend against dictionary attacks.

4.2.1 RSA oblivious pseudo-random function (OPRF)

Various techniques have been studied to solve the problem of encryption key falsification in the CE method. In 2013, Bellare [15] proposed a key generation scheme using RSA OPRF, which is a combination of OPRF designed by Naor and Reingold [22] in 1997 and RSA blind signature proposed by Bellare et al. [24] in 2003 and Chaum [23] in 1983. With RSA OPRF, the key is generated by communicating with the server. The RSA blind signature-based operations are repeatedly performed by the server and the user. $K \leftarrow G(H(M)^d \ mod \ n)$ is used to generate a secret key d known only to the key server. The advantage here is that the user cannot know the secret key a, and the key server cannot know the source data M. This makes it impossible for an attacker to forge an encryption key using a dictionary attack (Fig. 9).

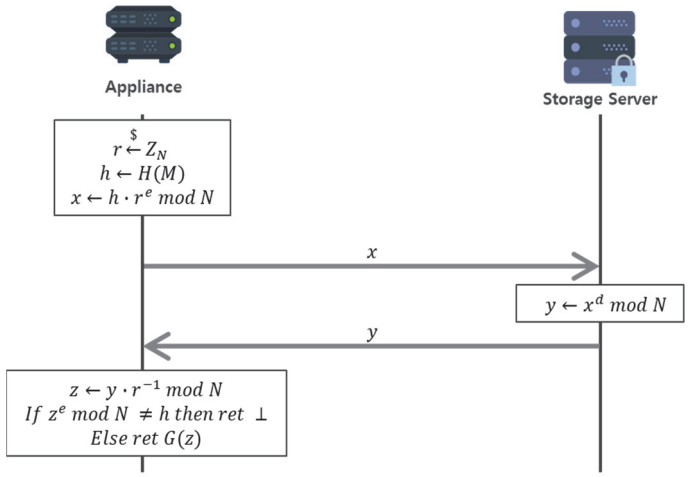

Fig. 9. RSA oblivious pseudo-random function.

4.2.2 Message locked encryption

Message locked encryption (MLE) is a data encryption technology proposed by Bellare et al. [15] in 2013. This technique proposed a method for generating the same encrypted data from the same data source. The general approach of MLE is to use the data source M to compute $K \leftarrow H(M)$, and use K as the key to generate $C \leftarrow E_K(M)$. MLE includes a total of four encryption technologies: CE, hash and CE without tag check (HCE1), hash and CE with tag check (HCE2), and randomized convergent encryption (RCE). CE and HCE1 only encrypt and decrypt data. HCE2 performs encryption and decryption together with integrity verification using tags. The RCE is based on HCE2. In the key generation process, the RCE generates a random key using a One Time Pad, performs encryption, and encrypts the random key with K. Therefore, the RCE has an advantage of improving the key generation entropy.

4.3. Prepare for Poison Attacks

Poison attacks occur when there is a mismatch between the source of secure data and the tags stored in the server. In other words, it occurs when the cipher data generated from data that is different from the source of the tag used when the user checks the duplication is uploaded. The damage caused by this is as follows:

- Loss of data source: If a user who has taken ownership by performing a subsequent upload on data where a poison attack occurred deletes the data source from the local storage, the source data cannot be acquired again.
- Damage due to modified data: Failure to detect data tampering during the download of data resulting from a poison attack can result in damage from malicious code contained in the modified data or property damage due to tampering with sensitive information.

Therefore, to prevent the poison attack, it is necessary to check whether the uploaded secure data is generated from the same source as the tag used in the check for duplicates.

4.4 Proof of Ownership

The proof of ownership technology has been proposed to defend against ownership forgery attacks. Typical proprietary forgery attacks occur when a user uploading data cannot verify that he owns the source of the data. Thus, various methods have been proposed to allow the owner of the data to verify ownership of the data source by performing a procedure to verify that the owner of the data owns the source.

4.4.1 PoWs (Proof of Ownership)

Merkle Tree, a hash tree proposed by Merkle [16], generates a single root node through the repeated concatenation and hashing of leaf nodes [17]. Merkle Tree has the concept of a sibling node. A sibling node is a neighboring node with the same parent. In Fig. 10, the sibling node of h_5 is h_6. In addition, the set of sibling node of each node in the path from the specific node to the root node is called a sibling path. A feature of Merkle Tree is that it is possible to create the same root node only if there is a sibling path for one leaf node even if b_1 to b_2 are not all known. Therefore, Halevi et al. [17] proposed the challenge-

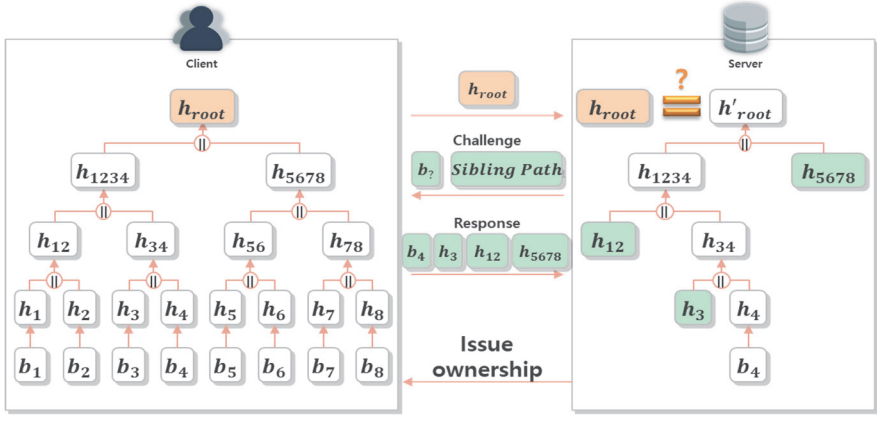

Fig. 10. Summary of PoWs.

response method of ownership verification using Merkle Tree. A feature of Merkle Tree is that the ownership of the data source can be verified by the data that can be generated without ownership of the data source. Therefore, when the user sends h_{root}, the root node generated, and the number of leaf nodes to the server, the server sends a randomly selected leaf node and sibling path to the user. The server can generate h'_{root} from the received data and judge ownership by determining whether it is equal to the h_{root} transmitted by the user.

This allows the user to verify that the data source is owned without providing the data source to the server. However, because a plurality of hash operations are performed for each data, the volume of computation increases drastically as the amount of data and the number of blocks increases.

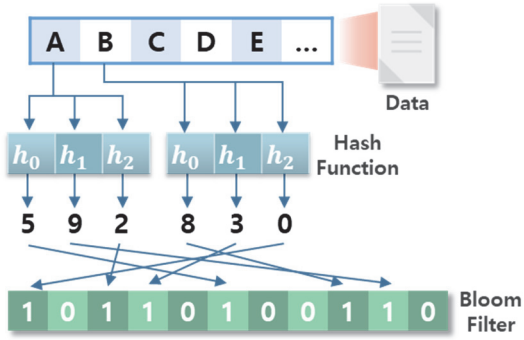

Fig. 11. Bloom filter verification.

4.4.2 bf-Pow

Bloom filter is a stochastic data structure designed by Bloom [25] as seen in Fig. 11. The hash value of the element to be included in the Bloom filter table and the output value is input as the index value of the Bloom filter table, and the value of the table corresponding to the index is replaced with 1. At this time, if the size of the bloom filter table is m, the index value range of the bloom filter table is 0 to m – 1. The hash algorithm is configured to output a value of 0 to m – 1 with a uniform probability. The hash algorithm should also be k different algorithms that output different results for the same input [16-18]. The process of adding and searching for elements in the Bloom filter is as follows:

- **Addition of elements**
  - i) Input the elements to be added to the Bloom filter into k different hash algorithms.
  - ii) Input the k hash values generated by the k hash algorithms into the bloom filter table indexes, respectively.
  - iii) Replace the corresponding value of the Bloom filter table searched for with the hash value as an index by 1.
- **Verification of elements**
  - i) Input the elements to be searched for in the bloom filter into k different hash algorithms.
  - ii) Input the k hash values generated by the k hash algorithms into the bloom filter table indexes, respectively.
  - iii) Make sure that the corresponding values in the Bloom filter table searched using the hash value as an index are all 1s.
  - iv) If the value of the searched index is all 1s, it is judged that the corresponding element is included

in the Bloom filter table. If the value output is 0, it is judged that the corresponding element is not included in the Bloom filter table.

According to the above procedure, the value corresponding to the hash value of the element is used as an index, the value of the corresponding position is replaced with 1, and 1 is continuously substituted even if the same index is repeatedly input. Conversely, to verify whether a specific element X is included in the bloom filter table, the bloom filter table is searched using the result obtained by inputting X into the same hash algorithm as an index. If any of the corresponding index values in the Bloom filter table is 0, the corresponding element X can be determined as an element not included in the Bloom filter table. However, the Bloom filter inherently has a positive error. There is a possibility that the value obtained by hashing the data A and the value obtained by hashing the data B exist. In this case, it can be determined that the data B exists in the storage through the Bloom filter value of the data A, assuming that the data A exists in the storage and the data B does not exist in the storage. Therefore, the proof of ownership through Bloom filter should be applied in a manner that reduces the probability of positive error. Blasco et al. [18] proposed a method of performing ownership verification using a bloom filter [8-10,17].

5. Secure Data Deduplication Systems

5.1 DupLESS

DupLESS is a cryptographic data deduplication system proposed by Bellare et al. [13] in 2013. The CE or MLE, which is mainly used in the deduplication of encrypted data, is fundamentally vulnerable to dictionary attacks, where the attacker can guess the plaintext data with only the encrypted data. DupLESS, on the other hand, additionally uses a key server to provide a secure cryptographic data deduplication system that is resistant to brute force attack.

The user generates the key K in cooperation with the key server via the RSA-OPRF protocol. In the process of communication between two entities, the user cannot know the private key as it is private to the key server, and the key server cannot know the message-based encryption key as it is secret information generated by the user. The actual key using RSA blind signature is $K \leftarrow G(H(M)^d \ mod\ n)$, where dd is the RSA private key of the key server, and G and H are the hash functions. The attacker cannot know the private key of the key server, and therefore cannot succeed in obtaining the plaintext file from the encrypted file through the offline preliminary investigation [42,43].

Users Alice and Bob generate cipher text $C_{1,A}, C_{1,B} \leftarrow E_K(M)$, upload it to the server, and remove duplicates from the server side. DupLESS encrypts the CE encryption key K using the secret key sk $(C_2 \leftarrow E_{sk}(K))$ unique to each user and stores it on the server.

DupLESS also protects against online brute force attacks by limiting the number of key request queries to the user's key server. In the worst case, if the attacker consults with the key server and knows the secret key information of the key server, data with low entropy becomes vulnerable to dictionary attacks, and the security of DupLESS is the same as the security of CE.

5.2 ClouDedup

Cloudedup is a cryptographic data deduplication system that provides block-level deduplication and

data confidentiality as proposed by Puzio et al. [26]. It is based on CE, but it defends against dictionary attacks by introducing a server that performs additional cryptographic functions. Unlike file-based deduplication, block-level redundancy eliminates the need to manage the CE keys for each block. In ClouDedup, the metadata manager performs key management and deduplication.

The client divides the file into blocks and encrypts each block with CE. The CE encryption key of the next block is encrypted with that of the previous block, and the remaining encrypted block key except for the first block key is encrypted again using the client secret key. It generates a signature value for the encrypted block. The client stores only the encryption key for the first block.

In the upload process, the server re-encrypts the encrypted block list, the encrypted block key list, and the signature value list of the blocks received from the client using the secret key of the server. It decrypts each block in the download process and verifies the signature value of the block using the user's public key.

The Metadata Manager (MM) stores the encrypted key and block signature values and removes duplicate blocks. To organize files from multiple blocks and manage file ownership, it manages file tables, pointer tables, and signature value tables as well as linked lists that include block ID and file ID.

The storage server stores the blocks received from the MM and does not play any role in deduplication. Therefore, it is very easy to construct a marine system using a commercial cloud storage service.

5.3 PerfectDedup

PerfectDedup is a data deduplication technology developed by Puzio et al. [26]. Other existing data deduplication technologies solve the conflict between data deduplication and data encryption by applying CE. However, such deduplication technologies encode all uploaded data while applying CE, which increases the computational overhead. Therefore, to address this issue, the popularity of the uploaded data is measured. CE is only applied if the data has a popularity higher than the threshold value.

PerfectDedup employs a perfect hash function for generating a unique ID associated with the data, prior to the data upload. The unique ID is then sent to the indexing server (IS) so that the IS can determine the data's popularity value. If the uploaded ID indicates data with low popularity such as privacy information, the data is encrypted by symmetric key encryption and is uploaded to the cloud service provider (CSP). However, if the data has high popularity, it is encrypted by CE and is then uploaded to the CSP. This allows PerfectDedup to perform data deduplication while providing data confidentiality to end users. Furthermore, PerfectDedup provides decreased accessibility to low-popularity data such as personal information, improves the accessibility of popular data, and thereby helps in reducing the computational overhead (Table 2).

Table 2. Comparison of secure data deduplication systems

	DupLESS	**ClouDedup**	**PerfectDedup**
Deduplication level	File level	Block level	Block level
Encryption algorithm	Convergent encryption	Convergent encryption	Convergent encryption
Key generate	Key server (RSA OPRF)	User	User
Brute-force attack resistibility	○	○	○
Dictionary attack resistibility	x	○	○
Additional features	-	-	Popularity-based

○=offer, x=not offer.

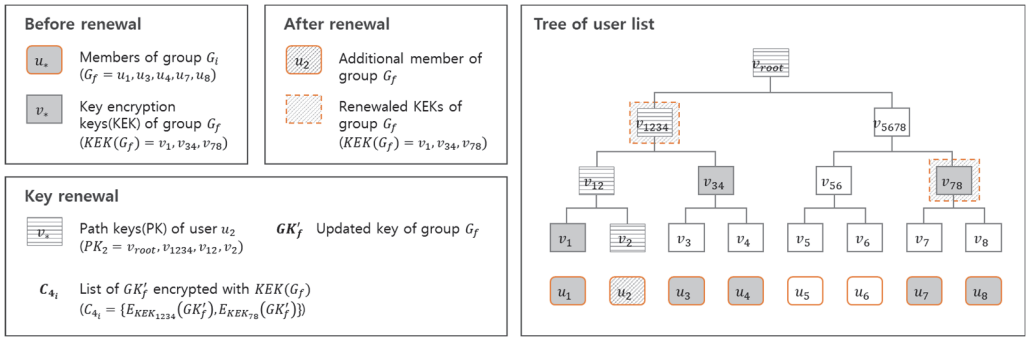

Fig. 12. Dynamic ownership management in Hur et al.'s scheme.

5.4. Hur et al.'s scheme

Hur et al.'s scheme [26] was proposed in 2016. The authors proposed a method for achieving dynamic ownership management in a secure data deduplication environment. This scheme addressed the problem of identifying the data owner using the ownership information of the user by providing anonymity of ownership. As a way to provide anonymity of ownership, a general method for confirming ownership through the ownership group was developed. However, this approach complicates ownership management because the ownership of the entire group must be changed at the time of ownership issue and renewal. Proxy re-encryption was proposed to solve this problem, along with a dynamic ownership management technology providing better efficiency. Fig. 12 shows how to create and update the ownership tree in this scheme.

6. Conclusion

Data deduplication technologies are designed to reduce storage costs. In recent years, there have been plans to reduce network traffic at the same time. Typically, data centers such as cloud storage are deployed as remote servers, and such remote servers should always be designed against data leakage because it is semi-trustworthy in that data can be leaked or transformed by insider or external factors at any time. In general, there are confidentiality and integrity requirements that can resist data leakage and tampering. Confidentiality can be solved by encrypting and storing the data. However, data deduplication technology is a technique that grasps the source of data and has a feature that conflicts with encryption technology that transforms the source of the data. Therefore, to encrypt data, encryption suitable for deduplication must be applied, and a typical technology for this is CE. However, even with the application of these technologies, additional research has been conducted while introducing new security threats.

In this paper, we have discussed techniques for performing data deduplication and techniques for performing secure data deduplication. These technologies are not completely redesigned but have some modified security technologies applied to the need for data deduplication. While these technologies have achieved their intended objectives, it is difficult to expect high levels of completeness while satisfying both computation and traffic efficiency due to the emergence of additional security threats and the use of additional security technologies to address them. However, the recent demand for privacy and secure technologies is expected to increase the demand and supply of secure data deduplication technologies in the future, which is expected to lead to more sophisticated technology research.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. NRF-2019R1A2C1085718) and the Republic of Korea's MSIT (Ministry of Science and ICT), under the High-Potential Individuals Global Training Program) (No. 2021-0-01516) supervised by the IITP (Institute of Information and Communications Technology Planning & Evaluation) and the BK21 FOUR (Fostering Outstanding Universities for Research) (No. 5199990914048).

References

[1] M. W. Storer, K. Greenan, D. D. Long, and E. L. Miller, "Secure data deduplication," in *Proceedings of the 4th ACM International Workshop on Storage Security and Survivability*, Alexandria, VA, 2008, pp. 1-10.

[2] J. R. Douceur, A. Adya, W. J. Bolosky, P. Simon, and M. Theimer, "Reclaiming space from duplicate files in a serverless distributed file system," in *Proceedings 22nd International Conference on Distributed Computing Systems*, Vienna, Austria, 2002, pp. 617-624.

[3] N. Kaaniche and M. Laurent, "A secure client side deduplication scheme in cloud storage environments," in *Proceedings of 2014 6th International Conference on New Technologies, Mobility and Security (NTMS)*, Dubai, UAE, 2014, pp. 1-7.

[4] A. Brinkmann, S. Effert, F. M. auf der Heide, and C. Scheideler, "Dynamic and redundant data placement," in *Proceedings of the 27th International Conference on Distributed Computing Systems (ICDCS)*, Toronto, Canada, 2007.

[5] A. Iyengar, R. Cahn, J. A. Garay, and C. Jutla, "Design and implementation of a secure distributed data repository," IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, 1998.

[6] A. W. Leung, E. L. Miller, and S. Jones, "Scalable security for petascale parallel file systems," in *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing*, Reno, NV, 2007, pp. 1-12.

[7] J. Li, M. N. Krohn, D. Mazieres, and D. E. Shasha, "Secure Untrusted Data Repository (SUNDR)," in *Proceedings of the 6th Symposium on Operating System Design and Implementation (OSDI)*, San Francisco, CA, 2004, pp. 121-136.

[8] E. L., Miller, D. D. Long, W. E. Freeman, and B. Reed, "Strong security for network-attached storage," in *Proceedings of the 1st UNENIX Conference on File and Storage Technologies (FAST),* Monterey, CA, 2002, pp. 1-13.

[9] S. Quinlan and S. Dorward, "Venti: a new approach to archival storage," in *Proceedings of the 1st UNENIX Conference on File and Storage Technologies (FAST),* Monterey, CA, 2002, pp. 89-101.

[10] C. Wang, Z. G. Qin, J. Peng, and J. Wang, "A novel encryption scheme for data deduplication system," in *Proceedings of 2010 International Conference on Communications, Circuits and Systems (ICCCAS)*, 2010, pp. 265-269.

[11] M. Miao, J. Wang, H. Li, and X. Chen, "Secure multi-server-aided data deduplication in cloud computing," *Pervasive and Mobile Computing*, vol. 24, pp. 129-137, 2015.

[12] J. Paulo and J. Pereira, "A survey and classification of storage deduplication systems," *ACM Computing Surveys (CSUR)*, vol. 47, no. 1, article no. 11, 2014.

[13] S. Keelveedhi, M. Bellare, and T. Ristenpart, "Dupless: server-aided encryption for deduplicated storage," in *Proceedings of the 22nd USENIX Security Symposium*, Washington, DC, 2013, pp. 179-194.

[14] K. Kim, K. Y. Chang, and I. K. Kim, "Deduplication technologies over encrypted data," *Electronics and Telecommunications Trends*, vol. 33, no. 1, pp. 68-77, 2018.

[15] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in *Advances in Cryptology – EUROCRYPT 2013*. Heidelberg, Germany: Springer, 2013, pp. 296-312.

[16] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Advances in Cryptology – CRYPT'87*. Heidelberg, Germany: Springer, 1987, pp. 369-378.

[17] 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*, Chicago, IL, 2011, pp. 491-500.

[18] J. Blasco, R. Di Pietro, A. Orfila, and A. Sorniotti, "A tunable proof of ownership scheme for deduplication using bloom filters," in *Proceedings of 2014 IEEE Conference on Communications and Network Security*, San Francisco, CA, 2014, pp. 481-489.

[19] L. Marques and C. J. Costa, "Secure deduplication on mobile devices," in *Proceedings of the 2011 Workshop on Open Source and Design of Communication*, Lisbon, Portugal, 2011, pp. 19-26.

[20] J. Xu, E. C. Chang, and J. Zhou, "Weak leakage-resilient client-side deduplication of encrypted data in cloud storage," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, Hangzhou, China, 2013, pp. 195-206.

[21] J. Li, X. Chen, M. Li, J. Li, P. P. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," *IEEE Transactions on Parallel And Distributed Systems*, vol. 25, no. 6, pp. 1615-1625, 2014.

[22] M. Naor and O. Reingold, "Number-theoretic constructions of efficient pseudo-random functions," *Journal of the ACM*, vol. 51, no. 2, pp. 231-262, 2004.

[23] D. Chaum, "Blind signatures for untraceable payments," in *Advances in Cryptology*. Boston, MA: Springer, 1993, pp. 199-203.

[24] M. Bellare, C. Namprempre, D. Pointcheval, and M. Semanko, "The one-more-RSA-inversion problems and the security of Chaum's Blind Signature Scheme," *Journal of Cryptology*, vol. 16, no. 3, pp. 182-215, 2003.

[25] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422-426, 1970.

[26] P. Puzio, R. Molva, M. Onen, and S. Loureiro, "ClouDedup: secure deduplication with encrypted data for cloud storage," in *Proceedings of 2013 IEEE 5th International Conference on Cloud Computing Technology and Science*, Bristol, UK, 2013, pp. 363-370.

[27] P. Puzio, R. Molva, M. Onen, and S. Loureiro, "Block-level de-duplication with encrypted data," *Open Journal of Cloud Computing (OJCC)*, vol. 1, no. 1, pp. 10-18, 2014.

[28] J. Hur, D. Koo, Y. Shin, and K. Kang, "Secure data deduplication with dynamic ownership management in cloud storage," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 11, pp. 3113-3125, 2016.

[29] J. Camenisch and G. Neven, "Simulatable adaptive oblivious transfer," in *Advances in Cryptology-EUROCRYPT2007*. Heidelberg, Germany: Springer, 2007, pp. 573-590

[30] 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, 2010.

[31] D. Russell, "Data deduplication will be even bigger in 2010," 2010 [Online]. Available from: https://www.gartner.com/en/documents/1297513/data-deduplication-will-be-even-bigger-in-2010.

[32] M. Dutch, "Understanding data deduplication ratios," 2008 [Online]. Available: https://www.snia.org/sites/default/files/Understanding_Data_Deduplication_Ratios-20080718.pdf.

[33] S. Rafaeli and D. Hutchison, "A survey of key management for secure group communication," *ACM Computing Surveys (CSUR)*, vol. 35, no. 3, pp. 309-329, 2003.

[34] C. Park, D. Hong, C. Seo, and K. Y. Chang, "Privacy preserving source based deduplication in cloud storage," *Journal of the Korea Institute of Information Security & Cryptology*, vol. 25, no. 1, pp. 123-132, 2015.

[35] J. Wang and X. Chen, "Efficient and secure storage for outsourced data: a survey," *Data Science and Engineering*, vol. 1, no. 3, pp. 178-188, 2016.

[36] N. Cook, D. Milojicic, and V. Talwar, "Cloud management," *Journal of Internet Services and Applications*, vol. 3, no. 1, pp. 67-75, 2012.

[37] S. A. El-Booz, G. Attiya, and N. El-Fishawy, "A secure cloud storage system combining time-based one-time password and automatic blocker protocol," *EURASIP Journal on Information Security*, vol. 2016, no. 1, article no. 13, 2016. https://doi.org/10.1186/s13635-016-0037-0

[38] J. Kim and S. Nepal, "A cryptographically enforced access control with a flexible user revocation on untrusted cloud storage," *Data Science and Engineering*, vol. 1, no. 3, pp. 149-160, 2016.
[39] M. I. Salam, W. C. Yau, J. J. Chin, S. H. Heng, H. C. Ling, R. C. Phan, G. S. Poh, S. Y. Tan, and W. S. Yap, "Implementation of searchable symmetric encryption for privacy-preserving keyword search on cloud storage," *Human-centric Computing and Information Sciences*, vol. 5, article no. 19, 2015. https://doi.org/10.1186/s13673-015-0039-9
[40] U. Habiba, R. Masood, M. A. Shibli, and M. A. Niazi, "Cloud identity management security issues & solutions: a taxonomy," *Complex Adaptive Systems Modeling*, vol. 2, no. 1, pp. 1-37, 2014.
[41] N. Singh and A. K. Singh, "Data privacy protection mechanisms in cloud," *Data Science and Engineering*, vol. 3, no. 1, pp. 24-39, 2018.
[42] Z. Guan, J. Li, Y. Zhang, R. Xu, Z. Wang, and T. Yang, "An efficient traceable access control scheme with reliable key delegation in mobile cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, article no. 208, 2016. https://doi.org/10.1186/s13638-016-0705-2
[43] N. Fotiou, A. Machas, G. C. Polyzos, and G. Xylomenos, "Access control as a service for the cloud," *Journal of Internet Services and Applications*, vol. 6, no. 1, pp. 1-15, 2015.
[44] K. Hashizume, D. G. Rosado, E. Fernandez-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," *Journal of Internet Services and Applications*, vol. 4, article no. 5, 2013. https://doi.org/10.1186/1869-0238-4-5
[45] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage," in *Financial Cryptography and Data Security*. Heidelberg, Germany: Springer, 2014, pp. 99-118
[46] W. K. Ng, Y. Wen, and H. Zhu, "Private data deduplication protocols in cloud storage," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, Trento, Italy, 2012, pp. 441-446.
[47] X. Jin, L. Wei, M. Yu, N. Yu, and J. Sun, "Anonymous deduplication of encrypted data with proof of ownership in cloud storage," in *Proceedings of 2013 IEEE/CIC International Conference on Communications in China (ICCC)*, Xi'an, China, 2013, pp. 224-229.
[48] J. Li, X. Chen, X. Huang, S. Tang, Y. Xiang, M. M. Hassan, and A. Alelaiwi, "Secure distributed deduplication systems with improved reliability," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3569-3579, 2015.
[49] Y. J. Shin, J. Hur, and K. Kim, "Security weakness in the proof of storage with deduplication," 2012 [Online]. Available: https://eprint.iacr.org/2012/554.pdf.

Won-Bin Kim **https://orcid.org/0000-0001-7772-6860**

He received an M.S. degree from the Department of Computer Science Engineering at Soonchunhyang University, Korea, in 2015. He is now a Ph.D. candidate in the Department of Software Convergence at Soonchunhyang University, Korea. His research interests include cloud storage security, cryptography, data deduplication, and data sharing.

Im-Yeong Lee **https://orcid.org/0000-0002-8856-0103**

He received a B.S. degree from the Department of Electronic Engineering at Hongik University, Korea, in 1981, and the M.S. and Ph.D. degrees from the Department of Communication Engineering at Osaka University, Japan, in 1986 and 1989, respectively. From 1989 to 1994, he was a senior researcher at ETRI (Electronics and Telecommunications Research Institute), Korea. Currently, he is a professor at the Department of Software Convergence, Soonchunhyang University, Korea. His research interests include cryptography, information theory, and computer & network security.