Public Audit and Secure Deduplication in Cloud Storage using BLS signature

Taek-Young Youn[1], Ku-Young Chang[1], Kyung Hyune Rhee[2], and Sang Uk Shin[2*]
[1]Electronics and Telecommunications Research Institute (ETRI), Republic of Korea
{taekyoung, jang1090}@etri.re.kr
[2]Dept. of IT Convergence and Application Engineering
Pukyong National University, Republic of Korea
{khrhee, shinsu}@pknu.ac.kr

Abstract

Although many researches have been done individually on each topic of secure deduplication and integrity auditing, the study of the combined model of these two functions was relatively few. In this paper, we propose a scheme to provide client-side deduplication for the encrypted data while simultaneously supporting public audit of data stored in the cloud. The proposed scheme performs the challenge-response protocol for secure deduplication and integrity audit using BLS signature based homomorphic linear authenticator, and supports public audit using the third party auditor. The proposed scheme satisfies all the basic security objectives and improves the problems of the existing schemes.

Keywords: Secure Deduplication, Public Audit, Cloud Storage, BLS signature

1 Introduction

Cloud storage services allow users to outsource their data to cloud storage servers and access them whenever and wherever they are needed. Due to its convenience, the use of cloud storage services has been widespread recently and its usage is increasing. Commercial cloud storage services such as Dropbox, Google Drive, and iCloud are used by many individuals and businesses.

However, when storing data on untrusted servers such as a cloud storage, users require the protection of their sensitive information, that is, the guarantee of data confidentiality. Servers, on the other hand, want to perform deduplication to improve storage space efficiency. According to EMC research, 75% of digital data is duplicates [7]. This fact raises a deduplication technology. This problem can be solved by a technique known as secure deduplication. For secure deduplication, in [9], Harnik et al. showed some attacks in the case of client-side deduplication which can lead to data leakage. Later, Halevi et al. [8] first introduced the proofs of ownership (PoW) scheme based on Merkle tree. In [11], Bellare et al. formalized the convergent encryption (CE), which is defined as message-locked encryption, and presented another scheme called DupLESS which is a server-aided encryption mechanism for deduplicated storage and ensures semantic security.

With the guarantee of confidentiality, users want to be assured that their data is stored in the cloud without being corrupted and accurately. Untrusted cloud servers are vulnerable to internal and external security threats, and uncontrolled cloud servers may try to hide some data loss incidents from users. More seriously, the server may deliberately discard the rarely accessed data owned by the end user in order to reduce cost and storage space. Therefore, the user wants to periodically check that his data is stored on

Research Briefs on Information & Communication Technology Evolution (ReBICTE), Vol. 3, Article No. 14 (November 15, 2017)
*Corresponding author: Dept. of IT Convergence and Application Engineering, Pukyong National University, Republic of Korea

the cloud server accurately and without corruption. To do this, it is required to perform the verification of integrity effectively without downloading the entire data. Considering only integrity auditing for data outsourced to cloud servers, a number of the proof of retrievability (POR) schemes [4, 10, 14, 16] and the provable data possession (PDP) schemes [1, 2, 6, 15] have been proposed.

Although individual studies on secure deduplication and integrity auditing have been actively conducted, there are relatively few studies on the combination of these two functions. The basic goal of the combined model is to have less overhead than a simple combination of secure deduplication and integrity auditing. In particular, this paper aims to improve the computational redundancy of the simple combined scheme, that is, the authentication tag for integrity audit is computed, and separately the authentication tag for performing the PoW protocol is computed in the secure deduplication process.

In this paper, we propose a scheme to achieve both secure deduplication and integrity auditing in a cloud environment. The proposed scheme provides a client-side deduplication for encrypted data, while simultaneously supporting public auditing of data stored in the cloud. The proposed scheme performs the PoW and integrity auditing using the homomorphic linear authenticator (HLA) based on BLS signature, and it supports public auditing using the TPA (Third Party Auditor). The proposed scheme satisfies the security objectives, and improves the problems of the existing schemes. Also, it is more efficient than the existing schemes in terms of the client-side computational overhead.

This paper is organized as follows. Section 2 describes related works. In Section 3, we propose a secure deduplication technique that supports integrity auditing based on BLS signature, and analyze it in Section 4. Finally, Section 5 is the conclusions.

2 Related works

Secure deduplication is interesting for both industrial and research community, so several secure deduplication schemes have been proposed. Harnik et al. [9] showed some attacks in the case of client-side deduplication which can lead to data leakage, and in [8], the concept of proof of ownership has been introduced to prevent such attacks. Later, Bellare et al. [11] formalized the convergent encryption, which is defined as message-locked encryption, and then they presented another scheme called DupLESS which is a server-aided encryption mechanism for deduplicated storage and ensures semantic security.

To support data integrity in the cloud, two concepts, Provable Data Possession (PDP) and Proof of Retrievability (POR) are introduced. Ateniese et al. [1] first introduced PDP for assuring that the cloud storage providers indeed possess the files without retrieving or downloading the whole 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 assures the cloud servers possess the target files, but also guarantees their full recovery [10]. Since then, a number of POR schemes [4, 14, 16] and PDP schemes [2, 6, 15] have been proposed.

A simple combination of public integrity auditing and secure deduplication does not efficiently deal with both the secure deduplication and integrity auditing, because achieving storage efficiency contradicts with the deduplication of authentication tags. In [17], public auditing with deduplication scheme was proposed which is based on polynomial-based authentication tags and homomorphic linear authentication tags. Each user has to generate the integrity tags, even for the file that already exists in the cloud. And the data is available in its plain form on the cloud-side. Li et al. [12] 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 fully trusted TPA.

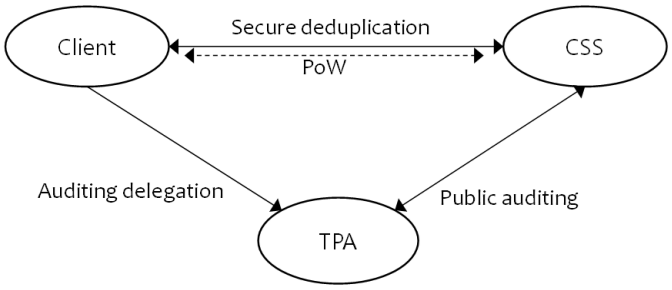

Figure 1: System model

3 The proposed scheme

3.1 System and security model

The proposed scheme utilizes the BLS signature-based Homomorphic Linear Authenticator (HLA) 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 (see Figure 1).

- Client (or user): Outsources data to a cloud storage. It uploads CE-encrypted data to the cloud storage to protect confidentiality. The client also want to be able to verify the integrity of the outsourced data. To do this, the TPA performs integrity auditing on behalf of the client.

- Cloud Storage Server (CSS): Provides data storage services to users. Deduplication technology is applied to save storage space and cost. We consider that the CSS may act maliciously due to insider/outsider attacks, software/hardware malfunctions, intentional saving of computational resources, etc [13]. During the deduplication process, the CSS carries out the PoW protocol to verify that the client owns the file. In addition, in the integrity audit process, it is necessary to generate and respond to a proof corresponding to the request of the TPA.

- TPA (Third Party Auditor): The audit work is delegated to the TPA in order to reduce the client's processing overhead. Instead of the client, the TPA sends a challenge to the CSS to periodically perform an integrity audit protocol. TPA is assumed to be a semi-trust, that is, an honest but curious model. It is assumed that it does not collude with other entities.

In this paper, we consider the following types of attacks: outside adversary, insider adversary CSS, and semi-honest adversary TPA.

- Outside adversary: Assuming 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 for the data, holding only the hash value of the data to deceive the CSS.

- Insider adversary CSS: The CSS assumes that it can act maliciously. It attempts to get information out of the user's encrypted data, and modify or delete the user data.

- Semi-honest adversary TPA: The TPA assumes to perform the protocol correctly, but in the process tries to obtain information about the user data.

Also, the proposed scheme should satisfy the following security objectives.

- Privacy: Except the information of duplication, no information about the outsourced data is leaked to the CSS and the TPA.
- Secure deduplication: Secure deduplication is supported without revealing any information except the information of duplication.
- Public verifiability: The TPA can verify the accuracy and availability of the outsourced data without querying the entire data and without intervention by the data owner.
- Storage correctness: If the CSS is keeping the user's data intact, it can pass the TPA's verification.

3.2 Detailed operation of the proposed method

The proposed scheme does not compute authentication values twice for a proof of PoW process and a proof of the integrity auditing, separately, but computes only one authentication value depending on the duplication. The proposed scheme uses the BLS signature based homomorphic authenticator [14] to generate the authentication value to provide secure deduplication and public integrity auditing. Let $e : G \times G \rightarrow G_T$ be a computable bilinear map with group G's support being $\mathbb{Z}_p$ for some large prime p. And g is a generator of G, and $BLSHash : \{0,1\}^* \rightarrow G$ is the BLS hash [3]. A user chooses a random $\alpha \xleftarrow{R} \mathbb{Z}_p$, and computes $v = g^{\alpha} (\in G)$. A user's private key is $sk = \alpha$, and the public key is $pk = v$. A user also generates a random key pair (spk, ssk) to digitally sign a file using a cryptographically secure signature scheme such as RSA PSS, DSA. It is assumed that the public key is securely distributed to the entities.

The proposed method consists of the following three procedures.

- First upload procedure: This is a case that a user first uploads a file that is not stored in the CSS. First, a file ID/Tag and a convergent encryption key K are generated, and the file is encrypted using CE with K and then uploaded to the CSS. The CSS stores the file owner, the file tag, and the ciphertext. The user computes an authentication tag for the integrity auditing and sends it to the TPA.
- Subsequent upload procedure: This is performed when a user attempts to upload a duplicate file already stored in the CSS. The CSS checks for the duplication using the file tag, and in the event of the duplication, proceeds with the PoW protocol to verify whether the user owns the file actually. If a user passes this process, the CSS adds the file ownership of the user to the stored file.
- Integrity auditing procedure: Periodic auditing is required to ensure that the files stored on the CSS are fully and intactly maintained. To reduce the user overhead, the TPA performs periodic integrity audits instead. To do this, the TPA first generates a random challenge and sends it to the CSS. The CSS responds by generating a corresponding proof using the stored file. And then the TPA verifies that the response is valid and completes the integrity audit.

3.2.1 First upload procedure

A client U_{ID} wants to upload a file F to CSS. To do this, he should first generate a convergent key K and a file ID T_F. This process can be performed using existing schemes such as DupLESS [11] and it is omitted in this paper. The client sends the Upload Req message to the CSS including $\{U_{ID}, T_F\}$. The CSS checks if a duplicated file exists using T_F. If so, the CSS performs the First upload procedure by sending the First upload Req message to the client. Then, the client computes the ciphertext CT by encrypting F with the convergent key K. $CT = Enc_K(F)$, where $Enc_k()$ is a symmetric key encryption

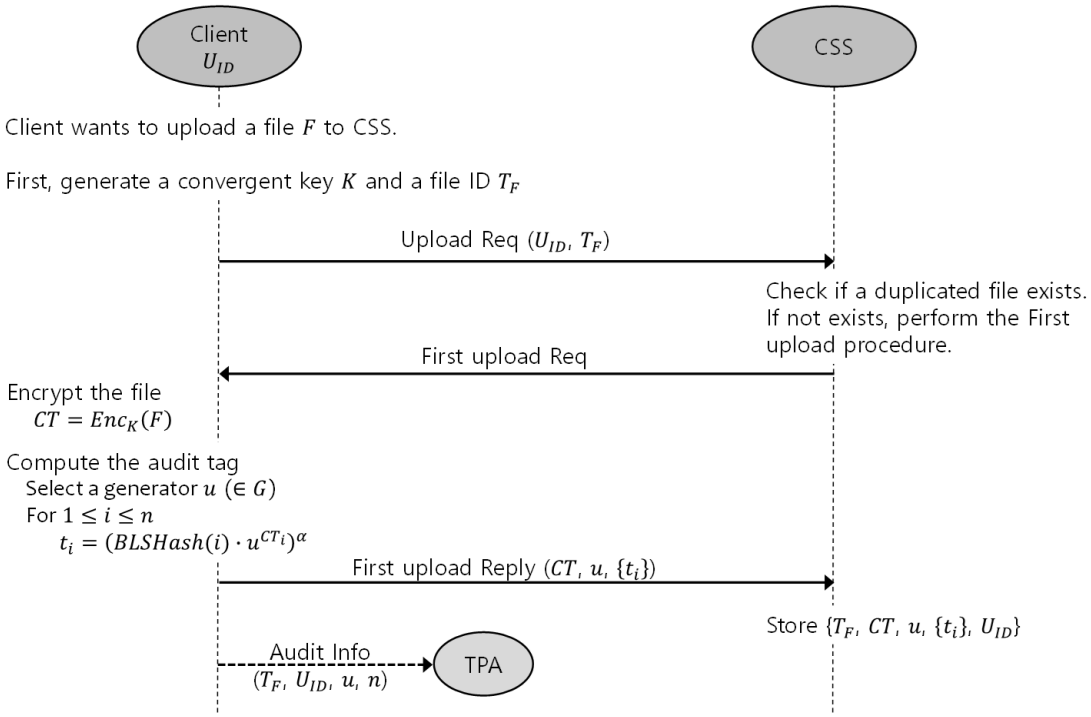

Figure 2: First upload procedure

mechanism with a key k such as AES in CTR mode [5]. The client also computes audit tags $\{t_i\}$ to periodically run the integrity audit procedure by the TPA. $t_i = \left(BLSHash(i) \cdot u^{CT_i}\right)^{\alpha}$, where CT_i is an i-th ciphertext block. The client replies $\{CT, \{t_i\}\}$ to CSS, and then CSS stores $\{T_F, CT, \{t_i\}, U_{ID}\}$. Also, the client sends the Audit Info message including $\{T_F, U_{ID}, u, n\}$ to TPA. The client keeps only (T_F, n) and deletes other information. Figure 2 shows the First upload procedure.

One consideration here is the management of the convergence key K. The convergence key K may be stored securely by each client individually. Alternatively, it can upload $ek = Enc_{mk}(K)$, encrypted using the secret master key mk of each client, to the CSS. In this case, the client only needs to keep mk secret. However, the CSS should store ek of each user for the duplicate file.

3.2.2 Integrity auditing procedure

On behalf of the client, the TPA periodically checks the integrity of the data stored in the CSS. To do this, the TPA first selects a random subset $I \in [1, n]$, and then selects v_i from $\mathbb{Z}_p$ randomly, for each $i \in I$. The challenge values for integrity auditing are $Q = \{I, (v_i)\}$. The TPA sends the Audit_Chall message with (T_F, Q) to the CSS. Then the CSS computes the proof values $\{\mu, \tau\}$ corresponding to the challenge as follows;

$$\mu \leftarrow \prod_{(i,v_i) \in Q} t_i^{v_i} \ (\in G), \quad \tau \leftarrow \sum_{(i,v_i) \in Q} v_i \cdot CT_i \ (\in \mathbb{Z}_p)$$

The CSS replies the Audit_Res message with $\{\mu, \tau\}$. The TPA verifies the proof by checking that the following holds;

$$e(\mu, g) == e\left(\prod_{i \in I} (BLSHash(i)^{v_i} \cdot u^{\tau}), v\right)$$

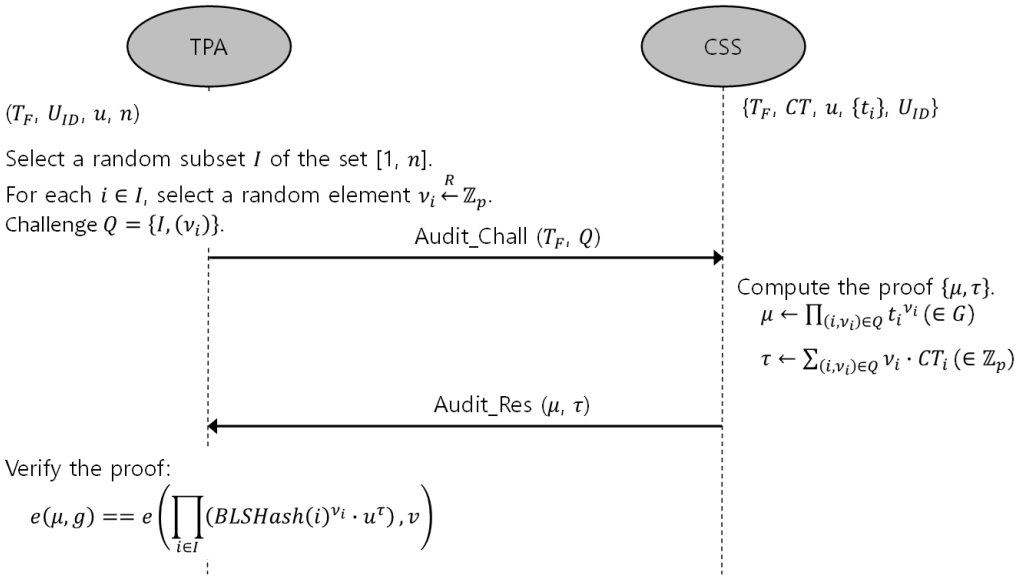

Figure 3: Integrity auditing procedure

Figure 3 shows the Integrity auditing procedure.

3.2.3 Subsequent upload procedure

If the CSS already stores the same file, the Subsequent upload procedure is performed for the deduplication (see Figure 4). To verify whether the client $U_{ID'}$ actually owns the file F, the CSS runs the PoW protocol. To do this, the CSS generates the challenge values $Q = \{I, (v_i)\}$, where I is a random subset of the set $[1,n]$ and v_i is a random element selected in $\mathbb{Z}_p$. And the CSS sends the PoW Chall message with (u,Q) to the client. Then the client derives the corresponding proof values $\{\mu,\tau\}$: computes $t_i' = \left(BLSHash(i) \cdot u^{CT_i}\right)^{\alpha'}$ for each $i \in I$, and then computes $\mu \leftarrow \prod_{(i,v_i)\in Q} t_i'^{v_i}$ $(\in G)$, $\tau \leftarrow \sum_{(i,v_i)\in Q} v_i \cdot CT_i (\in \mathbb{Z}_p)$. The client sends the PoW Res message with the proof $\{\mu,\tau\}$ to the CSS. Finally, the CSS verifies the proof by checking that the following holds;

$$e(\mu,g) == e\left(\prod_{i\in I}(BLSHash(i)^{v_i} \cdot \mu^{\tau}), v'\right)$$

Upon successful completion of the PoW procedure, the CSS can issue a receipt to the client by adding a user ID, $U_{ID'}$ and a reference to the file to the list of the corresponding file access rights in order to ensure the file ownership. The client can request the TPA to periodically check the integrity of the file using this receipt. If the CSS manages encrypted convergent keys, the client must upload $ek' = Enc_{mk'}(K)$, encrypted using its secret master key mk', to the CSS.

4 Analysis of the proposed scheme

The proposed scheme satisfies the security objectives mentioned above. From the privacy perspective, the proposed method outsources the encrypted ciphertext to the CSS using the convergent encryption key. In the integrity auditing process, the TPA can also partially obtain the information about the ciphertext

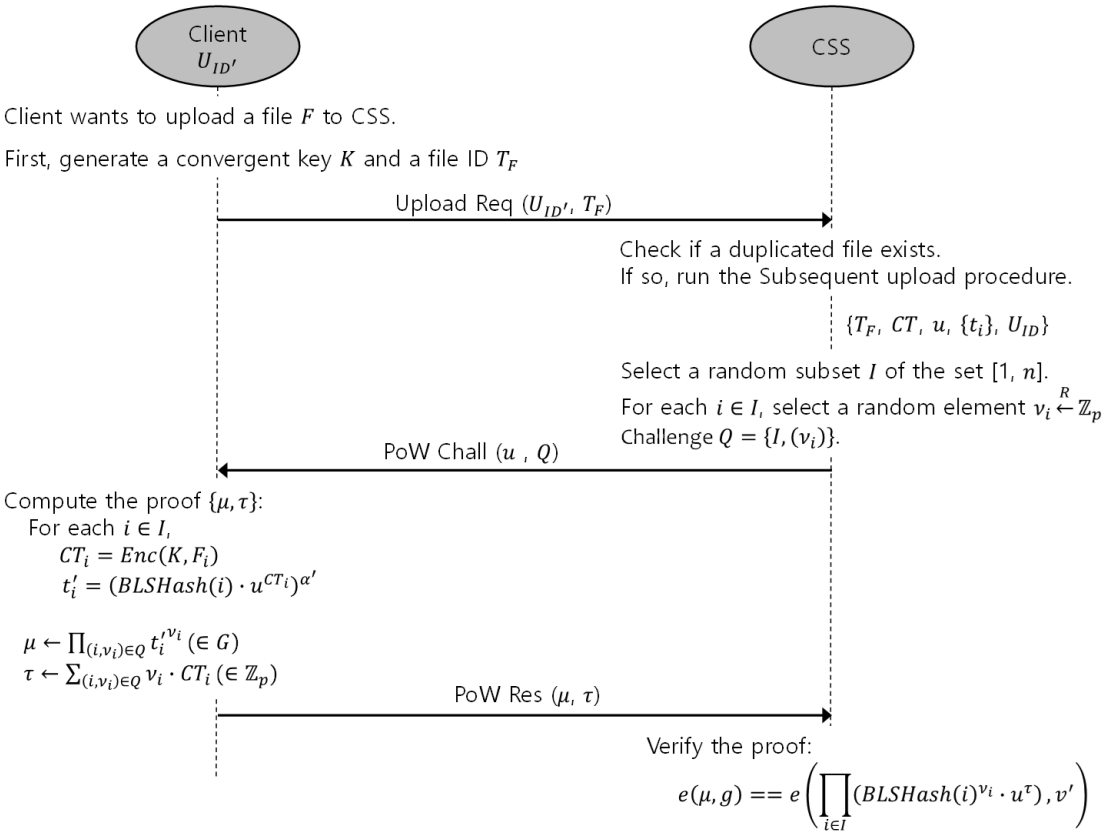

Figure 4: Subsequent upload procedure with PoW

only. Assuming that the convergent key generation process is performed through the OPRF (oblivious pseudo random function) protocol with the trusted key server as in DupLESS [11], it provides the security against offline brute-force attacks. An attacker who does not obtain the convergent key K can not get any information from the outsourced ciphertext, except the information of duplication. The proposed scheme also supports secure deduplication by providing the deduplication for the ciphertext. This also depends on the security of the convergent key, as mentioned above.

The TPA then audits the integrity of the data stored in the CSS without querying the entire data and without the user intervention. This depends on the security of the BLS-based homomorphic authentication tag, and its security is analyzed in [14]. Therefore, the proposed scheme supports public verifiability. Finally, if the CSS is keeping the data intact, it can pass the TPA's verification. This is also provided by the security of the BLS-based homomorphic authentication tag used, thus ensuring the correctness of the storage.

In comparison with existing schemes, first, [17] supports integrity auditing and deduplication using a polynomial-based authentication tag and a homomorphic linear tag. During the setup process, the user computes a homomorphic linear tag and uploads it to the cloud server along with the file. The TPA and the cloud server perform integrity auditing through the interaction using a polynomial-based authentication tag. In the deduplication process, when the cloud server randomly selects a set of block indexes for the PoW and sends them to the user, the user transmits the corresponding plaintext blocks as the respond. Then the cloud server verifies the file ownership by verifying the validity of the received blocks using the pairing operation. The biggest problem with this scheme is that the data is used as a plain text on the cloud side, so it does not support secure deduplication. Also, regardless of the file duplication,

Table 1: Comparison with existing schemes

	Secure deduplication	Privacy-preserving public auditing	Efficiency
[17]	X (because of available in its plain form on CSS)	O	Require two different authentication tags for PoW and for auditing, respectively, regardless of duplication.
[12]	O	X (because the file is uploaded to TPA)	Loss of bandwidth advantages in the client side
Proposed scheme	O	O	Require a computation of only one tag of both authentication tags, and ensure bandwidth advantages in the client side

users always have to compute authentication tags. This results in a high computational overhead on the user side.

In [12], the client uploads a file to a TPA that is assumed to be honest. This is a very strong assumption, since most of the TPA in the existing public auditing-related papers is assumed to be semi-honest. It also wastes the bandwidth on the client side because it always uploads the file to the TPA. When the client uploads the file to the TPA, the TPA computes a homomorphic signature for integrity auditing and uploads it to the cloud server along with the file.

The proposed scheme improves the problems of the above two methods. Table 1 shows the comparison with existing scheme. The TPA of the proposed scheme is assumed to be semi-honest, and also does not upload the file to the TPA. In addition, the proposed method supports deduplication and integrity auditing for the encrypted data. And the client only needs to perform a single authentication tag computation. It has similar computational overhead in each case of the first upload and of the duplicate upload. That is, in the case of the first upload, the authentication tag for the integrity audit is computed, and in the case of the deduplication, the authentication tag for the PoW is generated. Therefore, it is more efficient than the existing schemes in terms of the client-side computational overhead.

5 Conclusion

When storing data on untrusted servers such as cloud storage, users require the protection of their sensitive information. Additionally, users want to be assured that their data is stored in the cloud without being corrupted and accurately. So, in this paper, we proposed a scheme to achieve both secure deduplication and integrity auditing in a cloud environment. The proposed scheme provides a client-side deduplication for encrypted data, while simultaneously supporting public auditing of data stored in the cloud. We used the homomorphic linear authenticator based on BLS signature to perform the PoW and integrity auditing, and the proposed scheme supports public auditing using the TPA. The proposed scheme satisfied the security objectives, and improved the problems of the existing schemes. Also, it is more efficient than the existing schemes in terms of the client-side computational overhead.

Acknowledgments

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government. [17ZH1700, Develpoment of storage and search technologies over encrypted database]

References

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song. Provable data possession at untrusted stores. In *Proc. of the 14th ACM conference on Computer and communications security (CCS'07), Alexandria, Virginia, USA*, pages 598–609. ACM Press, October 2007.

[2] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik. Scalable and efficient provable data possession. In *Proc. of the 4th international conference on Security and privacy in communication netowrks (SecureComm '08), Istanbul, Turkey*, pages 1–10. ACM Press, September 2008.

[3] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. *Journal of Cryptology*, 17(4):297–319, September 2004.

[4] Y. Dodis, S. P. Vadhan, D. Wichs, et al. Proofs of retrievability via hardness amplification. In *Proc. of the 6th Theory of Cryptography Conference on Theory of Cryptography (TCC'09), San Francisco, California, USA, LNCS*, volume 5444, pages 109–127. Springer-Verlag, March 2009.

[5] M. Dworkin. Recommendation for block cipher modes of operation. methods and techniques. No. NIST-SP-800-38A. NATIONAL INST OF STANDARDS AND TECHNOLOGY GAITHERSBURG MD COMPUTER SECURITY DIV, 2001. `https://csrc.nist.gov/publications/detail/sp/800-38a/final` [Online; Accessed on October 3, 2017].

[6] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia. Dynamic provable data possession. In *Proc. of the 16th ACM conference on Computer and communications security (CCS'09), Chicago, Illinois, USA*, pages 213–222. ACM Press, November 2009.

[7] J. Gantz and D. Reinsel. The digital universe decade - are you ready? IDC White Paper, 2010. http://www.emc.com/collateral/analyst-reports/idc-digital-universe-are-you-ready.pdf.

[8] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of ownership in remote storage systems. In *Proc. of the 18th ACM conference on Computer and communications security (CCS'11), Chicago, Illinois, USA*, pages 491–500. ACM Press, October 2011.

[9] D. Harnik, B. Pinkas, and A. Shulman-Peleg. Side channels in cloud services: Deduplication in cloud storage. *IEEE Security & Privacy*, 8(6):40–47, November 2010.

[10] A. Juels and B. S. Kaliski Jr. Pors: proofs of retrievability for large files. In *Proc. of the 14th ACM conference on Computer and communications security (CCS'07), Alexandria, Virginia, USA*, pages 584–597. ACM Press, October 2007.

[11] S. Keelveedhi, M. Bellare, and T. Ristenpart. Dupless: server-aided encryption for deduplicated storage. In *Proc. of the 22nd USENIX Security Symposium (USENIX Security'13), Washington, D.C., USA*, pages 179–194. USENIX, August 2013.

[12] J. Li, J. Li, D. Xie, and Z. Cai. Secure auditing and deduplicating data in cloud. *IEEE Transactions on Computers*, 65(8):2386–2396, August 2016.

[13] X. Liu, W. Sun, H. Quan, W. Lou, Y. Zhang, and H. Li. Publicly verifiable inner product evaluation over outsourced data streams under multiple keys. *IEEE Transactions on Services Computing*, 10(5):1–14, February 2016.

[14] H. Shacham and B. Waters. Compact proofs of retrievability. In *Proc. of the 14th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'08), Melbourne, Australia, LNCS*, volume 5350, pages 90–107. Springer-Verlag, December 2008.

[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 Transactions on Parallel and Distributed Systems*, 22(5):847–859, October 2011.

[16] J. Yuan and S. Yu. Proofs of retrievability with public verifiability and constant communication cost in cloud. In *Proc. of the 2013 international workshop on Security in cloud computing (Cloud Computing'13), Hangzhou, China*, pages 19–26. ACM Press, May 2013.

[17] J. Yuan and S. Yu. Secure and constant cost public cloud storage auditing with deduplication. In *Communications and Network Security (CNS), 2013 IEEE Conference on, National Harbor, Maryland, USA*, pages 145–153. IEEE, October 2013.

Author Biography

Taek-Young Youn received his BS, MS, and Ph.D from Korea University in 2003, 2005, and 2009, respectively. He is currently a senior researcher at Electronics and Telecommunications Research Institute (ETRI), Korea. His research interests include cryptography, information security, and data privacy.

Ku-Young Chang received his B.S., M.S. and Ph.D. degrees in mathematics from Korea University, Seoul, Korea on 1995, 1997, and 2000, respectively. He is currently a principal researcher of Cryptography Research Section at Electronics and Telecommunication Research Institute, Daejeon, Korea. His research interests include cryptography, data privacy, and finite field theory.

Kyung-Hyune Rhee received his M.S. and Ph.D. degrees from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea in 1985 and 1992, respectively. He worked as a senior researcher in Electronic and Telecommunications Research Institute (ETRI), Daejeon, Korea from 1985 to 1993. He also worked as a visiting scholar in the University of Adelaide in Australia, the University of Tokyo in Japan, the University of California at Irvine in USA, and Kyushu University in Japan. He has served as a Chairman of Division of Information and Communication Technology, Colombo Plan Staff College for Technician Education in Manila, the Philippines. He is currently a professor in the Department of IT Convergence and Application Engineering, Pukyong National University, Busan, Korea. His research interests center on multimedia security and analysis, key management protocols and mobile ad-hoc and VANET communication security.

Sang Uk Shin received his M.S. and Ph.D. degrees from Pukyong National University, Busan, Korea in 1997 and 2000, respectively. He worked as a senior researcher in Electronics and Tele-communications Research Institute, Daejeon Korea from 2000 to 2003. He is currently a professor in Department of IT Convergence and Application Engineering, Pukyong National University. His research interests include digital forensics, e-Discovery, cryptographic protocol, mobile and wireless network security and multimedia content security.