International Journal of Engineering & Technology, 7 (4) (2018) 2473-2476

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.15742
Research paper

An improved strong key exposure resilient auditing for cloud storage auditing

R. Ahila [1] *, Dr. S. Sivakumari [2]

[1] Assistant Professor, Department of Information Technology, School of Engineering, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore- 641 108
[2] Professor and Head Department of Computer Science and Engineering, School of Engineering, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore- 641 108
**Corresponding author E-mail: ahilaph.d@gmail.com*

Abstract

One of the most essential services of cloud computing is cloud storage. For cloud storage auditing, key exposure is a serious security problem which is solved by updating client secret keys regularly. However, this leads to local burdens to clients. A cloud source auditing with verifiable outsourcing of key updates paradigm was used to make the key updates as transparent for the client where Third Party Auditor (TPA) was played the role of authorized party. It holds only an encrypted version of client's secret key. However, key exposure security problem is not fully solved by this scheme. So in this paper, improved strong key exposure resilient auditing is introduced to solve the key exposure security problem and improve the security of cloud storage. In the proposed paradigm, a novel key update technique is used where update message is created by TPA and it sends to the client. The client updates their signing secret key based on the update message and client's private key. Moreover, public key is obtained from the client while updating a message which improves the security of cloud storage. Thus this scheme makes the malicious cloud unable to get the signing secret key in unexposed time periods.

Keywords:*Cloud Computing; Cloud Storage Auditing; Improved Strong Key Exposure Resilient Auditing; Key Exposure; Third Party Auditing.*

1. Introduction

Cloud Computing [1] provides the users a path by which they can easily access all the applications and lot of data or content globally available. There are different security issues in cloud environment [2]. Cloud storage auditing [3, 4] is one of the significant security techniques in cloud storage that is used to verify the integrity of the data stored in public cloud. In cloud storage auditing process [5], Third Party Auditor (TPA) [6, 7] is a kind of inspector. There are private audit ability and public audit ability. Even though, private audit ability can achieve highest scheme efficiency, public audit ability allows anyone, not just the client to challenge the cloud server for the correctness of data storage while keeping no private information. To let off the burden of management of data of the data owner, TPA will audit the data of client. It audits or checks the data integrity in the cloud. While auditing the cloud storage, there may be a chance of exposing the secret key of the client, it would leads to forging the later when the client requests for the same. There are several reasons for key exposure. Some of them are key management, internet based security attacks and trading with hackers.

In cloudthe clients or TPA performed storage auditing with verifiable outsourcing of key updates auditing scheme [8] the key update operations for cloud storage auditing. Here, TPA was considered as authorized party and they hold encrypted version of client secret key. It was updated in encrypted state in each time period. When a client needs to upload a file in cloud, they have to download an encrypted client secret key from authorized party and they need to decrypt to upload. In addition to this, the validity of the encrypted secret key was verified by the client. But, the problem of key exposure was not fully solved by this scheme. In order to effectively solvethe problem of key exposure for cloud auditing, a strong key exposure resilient auditing scheme is designed in this paper. It is designed in such a way that the key exposure in one time period doesn't affect the security of cloud storage auditing in other time periods. In the proposed scheme, the public key is obtained from the client at each time while updating a message. This improves the cloud storage security effectively. The experimental shows that the proposed key exposure resilient auditing scheme is secure and efficient than the other schemes.

2. Literature review

A secure storage, verification and auditing (secSVA) architecture [9] was proposed to provide secure third party auditing in cloud environment.It supports data authentication, verification, auditing, integrity and confidentiality for cloud storage. It provided an attribute based security framework with secure deduplication for big data storage in the cloud. A Kerberos based identity verification and authentication scheme for cloud based big data storage was presented in this framework. In addition to that, a data integrity scheme with trusted party auditing using a Merkle hash tree scheme was also presented in this framework.

A novel public auditing scheme [10] was proposed for secure cloud data storage based on Dynamic Hash Table (DHT). DHT was a two dimensional data structure placed at a TPA. It supports dynamic data auditing by recording the data property information. To minimize the computational cost and the communication overhead, the proposed public auditing scheme migrate the authorized information from the cloud service provider to the TPA. In addi-

tion to that, the proposed scheme was extended to support the privacy preservation through the combination of homomorphic authenticator based on the public key with the random masking generated by the TPA. It also supported the batch auditing by introducing the aggregate BLS signature technique. But this scheme is less effective.

For secure auditing, a novel public auditing scheme called as multi-replica dynamic public auditing (MuR-DPA) scheme [11] was proposed. Based on the Merkle Hash Tree (MHT) a novel authenticated system was combined with in the public auditing scheme. This scheme was named as MR-MHT. In the computation of MHT nodes, levels and ranks of each nodes were appended which helped to support full dynamic data updates and authentication of block indices. The level values of nodes in MR-MHT were arranged in a manner of top down order. Furthermore, for each data block all replica blocks were arranged into a same replica sub tree. Hence, such a configuration allowed efficient verification of updates for multiple replicas. However, the size of the dataset was greatly influence proof size.

A public auditing scheme [12] was proposed to regenerate-code-based cloud storage.A proxy was introduced in the proposed public auditing scheme to resolve the regeneration problem of failed authenticators. This proxy was confidential to regenerate the authenticators into the traditional public auditing system model. In addition to that, a novel public verifiable authenticator was created which generated by a couple of keys and regenerated by using partial keys. Hence, this scheme was completely release data owners from online burdens. Moreover, the encode coefficients were randomized with a pseudorandom function which preserved the privacy of data.

A cloud storage auditing protocol [13] with built-in key-exposure resilience was designed. It reduced the damage of the client key exposure in cloud storage auditing. The integrity of data which was initially stored in the data was verified by the proposed cloud storage auditing protocol even if current secret key of client was exposed. A practical solution for the damage of the client key exposure was introduced in the cloud storage auditing only after formalization of definition and the security model of auditing protocol with key resilience. The proposed protocol was secure and efficient.

3. Materials and methods

In this section, the proposed a strong key exposure resilient auditing is described in detail. In the existing cloud storage auditing with verifiable outsourcing of key updates, an encrypted client secret key from authorized party was downloaded by client when they need to upload a file in cloud and then they have to decrypt to upload. There may be a chance for disclosing the client secret key when a same client secret key was maintained. This is the major of the cloud storage auditing with verifiable outsourcing of key updates. So the problem of key exposure is solved by proposed key exposure resilient auditing. The main objective of proposed method is to design a scheme in such a way that the signing secret keys changes in different time periods while the public key is unchanged in all of time periods to improve the security of data.

3.1. Strong key exposure resilient auditing

By making the signing secret key in each time period be a multiplication of two parts, the strong key-exposure resilience is achieved. Each part is the power of $F_1(x)$, where x denotes the current time period and F represents a hash function. One part is the update message generated by the authenticators, which is computed by the current time period and the secret key of the TPA. Another part is computed from the secret key of the client and current time period. It needs to be concentrates that the signing secret key in any time period must be jointly created by the TPA and client. Hence, the proposed scheme supports both the efficient key update and provable security. The property of blackness verifiability and the structure of the signing secret keys is supported by the designed authenticator.

3.2. Description of strong key exposure resilient auditing scheme

Consider $\hat{m}: Y_1 \times Y_2$ be a bilinear map, where Y_1 and Y_2 are two multiplicative groups with order p. Consider, s and t are two generators of group Y_1 and $F_1: \{0,1\}^* \to Y_1, F_2 \to \{0,1\}^* \times Y_1 \to Y_1$ are two cryptographic hash functions. A digital signature called as Sig is used to ensure the integrity of the file identifier name and time period x. Assume (spk, ssk) is a pair of public key and secret key corresponding to signature Sig, the client has held the secret key and the public key has been published. Such an assumption can simplify the proposed scheme description thereafter. Initially, the client divides the one file X stored on the cloud into a set of t ordered blocks$b_1, b_2, \dots b_t$, whereZ_p^*. In time period x, the signing secret key of the client is $W_x \in Y_1$. The authenticator for each block $b_i$in time period x is generated as follows. The client selects a random $n \in Z_p^*$ and calculates$N = s^n$. They compute the authenticator for each block b_i in time period x as $\delta_i = F_2(x||i||name, N)^n . u^{nb_i} . W_x$, where name denotes the name of the file X.

3.2.1. Sys setup algorithm

a) Client randomly selects a private key$W_c \in Z_p^*$. $T_c = s^{W_c}$is the client's public key.
b) Client randomly selects $W_{TPA} \in Z_p^*$ and it send to the TPA as the TPA's secret key. $T_{TPA} = s^{W_{TPA}}$denotes the TPA's public key.
c) Set the system public key $T = (s, t, T_{TPA}, T_c, spk)$.UMGen algorithm From the client, get public key T at each time period x.

At the beginning of the time period x, TPA computes the update message $\sigma_x = F_1(x)^{W_{TPA}}$ to the client according to their secret key W_{TPA}. TPA sends the update message σ_x to the client.

By the using following equation, the client can able to verify the update message as valid or invalid.

$$\hat{m}(s, \sigma_x) = \hat{m}(T_{TPA}, F_1(x)) \quad (1)$$

3.2.2. Ckey update algorithm

The client compute $W_x = F_1(x)^{W_c}$ after getting the update message σ_x from the TPA at the beginning time period x. σ_x is the signing secret key in time period x.

3.2.3. Authgenalgorithm

When the client wants to upload a file $X = \{b_i\}$ $(1 \le i \le t)$ to the cloud in time period x, they does as follows:

a) The client selects $n \in N^{Z_p^*}$ and then computes $N = s^n$.
b) Then the client computes the authenticators $\delta_i = F_2(x||i||name, N)^n . u^{nb_i} . W_x$, where name denotes the name of the file.
c) The client uploads the file tag $tag = name||x||Sig_{ssk}(name||x)$ and the set of authenticators $\Phi = \{x, N, \delta_i, \dots, \delta_t\}$ along with the file X to the cloud.

3.2.4. Proof gen algorithm

a) The TPA first verifies the validity of the file tag by checking whether $Sig_{ssk}(name||x)$ is a valid signature using spk. If it is, then the TPA selects a c-element subset $Sub = \{s_1, s_2, \dots, s_c\}$ of set [1,t] as the index of the blocks to be checked.
b) For each $i \in Sub$, the TPA selects the random values $u_i(||u_i|| < ||p||)$, and sends the challenge $\{i, u_i\}_{i \in Sub}$to the cloud.

c) When the cloud receives the challenge $\{i, u_i\}_{i\in Sub}$, they compute an aggregate authenticator $\delta = \prod_{i\in Sub} \delta_i^{u_i}$. They also computes $\beta = \sum_{i\in Sub} b_i u_i$.
d) The cloud sends $P = (x, N, \delta, \beta)$ as their reply.

3.2.5. Proof verify algorithm

When the TPA receives the proof P, verifies whether the following equation holds:

$$\hat{m}(s,\delta) = \hat{m}\left(N, \prod_{i\in Sub} F_2(x||i||name, N)^{u_i}.u^{\beta}\right)\hat{m}\left(T_c, T_{TPA}, F_1(x)^{\sum_{i\in Sub} u_i}\right)$$

If it holds, then return true, else return false.

4. Results and discussion

In this section, the performance of the existing and proposed cloud storage auditing schemes are analyzed in terms of time for the client to update the secret key, communication overhead and time in an auditing process. With the help of the Pairing-Based Cryptography (PBC) library [14], the proposed cloud storage auditing scheme is evaluated.

4.1. Time for client to update the secret key

The time for client to update the secret key calculates the time to update key on client side by using existing and proposed cloud storage auditing scheme. The following Table1 shows the comparison between existing and proposed cloud storage auditing scheme in terms of time for client to update the secret key.

Table 1: Comparison of Time for Client to Update the Secret Key

Current Time Period	CSA-VOKU	ISKERA
2	0.91	0.8
4	0.94	0.85
6	0.92	0.81
8	0.93	0.86
10	0.91	0.82
12	0.9	0.8
14	0.92	0.83

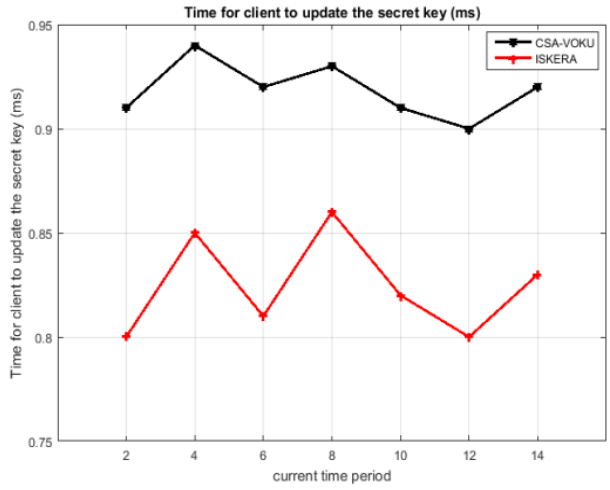

Fig. 1: Time for Client to Update the Secret Key.

Fig.1 shows the comparison between existing Cloud Storage Auditing with Verifiable Outsourcing of Key Updates (CSA-VOKU) and proposed Improved Strong Key-Exposure Resilient Auditing (ISKERA) scheme in terms of time for client to update the secret key. X axis denotes the current time period and Y axis denotes the time for client to update the secret key in terms of milliseconds. When the time period is 10, the time for client to update the secret key using ISKERA is 9.9% lesser than CSA-VOKU. From the Fig.1 it is proved that the proposed ISKERA scheme has less time for client to update the secret key than the existing scheme.

4.2. Communicational overhead

The communicational overhead is the proportion of time spent on communicating between TPA and client. The following Table2 shows the comparison between existing and proposed cloud storage auditing scheme in terms of communicational overhead.

Table 2: Comparison of Communicational Overhead

Number of challenged blocks	CSA-VOKU	ISKERA
2	5.2	3.1
4	9	7.3
6	13.8	10.7
8	18	15
10	22	20

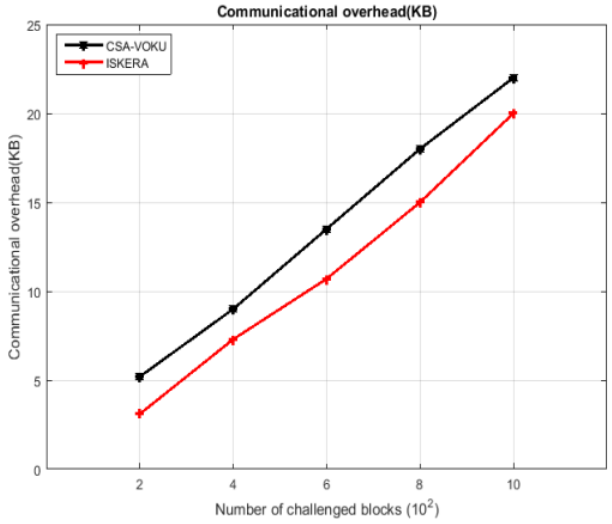

Fig. 2: Communicational Overhead.

Fig.2 shows the comparison of communicational overhead between existing CSA-VOKU and proposed ISKERA scheme. X axis denotes the number of challenge blocks and Y axis denotes the communicational overhead in terms of KB. When the number of challenged blocks is 800, the communicational overhead by ISKERA is 16.7% lesser than CSA-VOKU. From the Fig.2, it is proved that the proposed ISKERA scheme has less communicational overhead than the existing scheme.

4.3. Time in auditing process

Time in auditing process is defined as amount of time taken for cloud storage auditing. The following Table3 shows the comparison between existing and proposed cloud storage auditing scheme in terms of time in auditing process.

Table 3: Comparison of Time in Auditing Process

Number of challenged blocks	CSA-VOKU-challenge gen	CSA-VOKU-proof gen	CSA-VOKU-proof ver	ISKERA-challenge gen	ISKERA-proof gen	ISKERA-proof ver
2	0.42	0.45	1.2	0.3	0.4	1
4	0.45	0.5	2.5	0.34	0.47	2.2
6	0.48	1	3.6	0.39	0.8	3.3
8	0.51	1.4	5	0.42	0.9	4.7
10	0.54	1.8	6	0.45	1	5.8

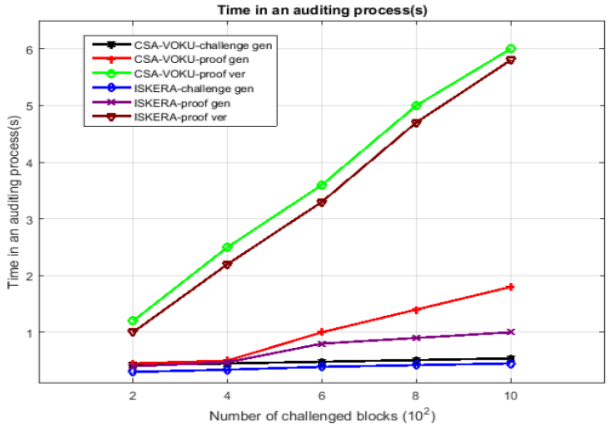

Fig. 3: Time in Auditing Process.

Fig.3 shows the comparison of time in auditing process between existing CSA-VOKU and proposed ISKERA scheme. X axis denotes the number of challenge blocks and Y axis denotes the time in auditing process in terms of seconds. From the Fig.3 it is proved that the proposed ISKERA scheme has less time in auditing process than the existing scheme.

5. Conclusion

This paper handles the problem of key exposure in cloud storage auditing by strong key exposure resilient auditing scheme. The integrity of files stored in the cloud is periodically checked by TPA authenticators. It helps to update client secret key by providing update message to client in different time interval. Moreover, in this scheme the security of the cloud storage auditing not only earlier than but also latter than the key exposure can be preserved. Thus the proposed scheme provides more security to the cloud storage. The experimental results show that the proposed cloud storage auditing scheme performs better than the existing cloud storage auditing scheme in terms of time for client to update the secret key, communicational overhead and time in auditing process.

References

[1] Singha S & Satav SD (2016), An Effective Approach for Key Exposure Resistance in Cloud using De Duplication and Tile Bitmap Method, *International Journal of Computer Applications*, 146(9), pp. 41-44.

[2] Ahila R & Sivakumar S (2017), A Survey on Security Issues in Cloud Environment, *International Journal of Research in Electronics and Computer Engineering*, 5(2), pp. 138-140.

[3] Yu Y, NiuL,Yang G, Mu Y, Susilo W (2014), On the security of auditing mechanisms for secure cloud storage, *Future Generation Computer Systems*, 30, pp. 127-132.https://doi.org/10.1016/j.future.2013.05.005.

[4] Yuan J, & Yu S (2013), Secure and constant cost public cloud storage auditing with deduplication. In *IEEE Conference onCommunications and Network Security (CNS)*, pp. 145-153. https://doi.org/10.1109/CNS.2013.6682702.

[5] Singha S, Satav SD (2015), A Survey Paper on Cloud Storage Auditing with Key Exposure Resistance, *International Journal of Science and Research(IJSR)*,4(11), pp. 1821-1823.

[6] Govinda K, Gurunathaprasad V, &Sathishkumar H (2012), Third party auditing for secure data storage in cloud through digital signature using RSA. *International Journal of Advanced Scientific and Technical Research*, 4(2), pp. 525-530.

[7] Mohta A, &Awasti LK (2012), Cloud data security while using third party auditor. *International Journal of Scientific & Engineering Research*, *3*(6), pp. 1-4.

[8] Yu J, Ren K, Wang C (2016), Enabling cloud storage auditing with verifiable outsourcing of key updates, *IEEE Transactions on Information Forensics and Security*, 11(6), pp. 1362-1375.https://doi.org/10.1109/TIFS.2016.2528500.

[9] Aujla GS, Chaudhary R, Kumar N, Das AK, Rodrigues JJ (2018), SecSVA: Secure Storage, Verification, and Auditing of Big Data in the Cloud Environment, *IEEE Communications Magazine*, 56(1), pp. 78-85.https://doi.org/10.1109/MCOM.2018.1700379.

[10] Tian H, Chen Y, Chang CC, Jiang H, Huang Y, Chen Y, Liu J (2015), Dynamic-hash-table based public auditing for secure cloud storage, *IEEE Transactions on Services Computing*, pp. 1-14.

[11] Liu C, Ranjan R, Yang C, Zhang X, Wang L, Chen J (2015), MuR-DPA: Top-down levelled multi-replica merkle hash tree based secure public auditing for dynamic big data storage on cloud, *IEEE Transactions on Computers*, 64(9), pp. 2609-2622.https://doi.org/10.1109/TC.2014.2375190.

[12] Liu J, Huang K, Rong H, Wang H, Xian M (2015), Privacy-preserving public auditing for regenerating-code-based cloud storage. *IEEE transactions on information forensics and security*, 10(7), pp. 1513-1528.https://doi.org/10.1109/TIFS.2015.2416688.

[13] Yu J, Ren K, Wang C, Varadharajan V (2015), Enabling cloud storage auditing with key-exposure resistance, *IEEE Transactions on Information forensics and security*,10(6), pp. 1167-1179.https://doi.org/10.1109/TIFS.2015.2400425.

[14] Lynn B, The pairing-based cryptography library, Internet: crypto. stanford. edu/pbc/[Mar. 27, 2013].