On Security of an Identity-Based Dynamic Data Auditing Protocol for Big Data Storage

Xiong Li, Shanpeng Liu, Rongxing Lu, *Senior Member, IEEE*, Xiaosong Zhang

***Abstract*—In this paper, we point out the security weakness of Shang *et al.*'s identity-based dynamic data auditing protocol for big data storage [IEEE Transactions on Big Data, 2019, doi: 10.1109/TBDATA.2019.2941882]. Specifically, we identify that their protocol is vulnerable to a secret key reveal attack, *i.e.*, the service provider (SP) can reveal the secret key of the data owner (DO) from the stored data. Further, SP can also generate a proof to pass the challenge of TPA (third party auditor) even if all block and tag pairs have been deleted. We hope that by identifying these design flaws, similar weaknesses can be avoided in future designs.**

***Index Terms*—Cloud storage, Auditing protocol, Dynamic audit, Private key reveal attack**

I. Introduction

Cloud storage, as it can provide users with efficient, secure and low-cost storage services without having to build a storage platform by themselves, has become a popular application along with the wide spread of cloud computing. However, as data users lose physical control over their data, the security of the outsourced data has attracted researchers' considerable attention. Among all security issues of cloud storage, data integrity is the most basic one, as it convinces data users that the data they store on the service provider is complete. To deal with the data integrity issue, the concept of data integrity auditing has been proposed, which usually adopts a third party auditor to audit whether the service provider has stored the users' data intact. Up to now, many public data integrity auditing protocols have been proposed by researchers to achieve different security and functional features, such as privacy preserving, storage correctness, batch verification, and dynamic support. However, most of solutions are designed based on the public key infrastructure (PKI), which however brings a heavy key management burden to the data users. As a result, identity-based cryptography has been exploited by researchers in the public auditing protocols to avoid the potential key management problem. For example, Wang *et al.* [1] and Yu *et al.* [2] have respectively proposed their identity-based public auditing protocols for cloud storage. Quite recently, Shang *et al.* [3] first proposed an identity-based public auditing protocol with dynamic support. However, in this paper, we point out Shang *et al.*'s protocol [3] is vulnerable to a private key reveal attack that the service provider can reveal the data owner's private key, and then the service provider can forge valid proof to pass the challenge of third-party auditor without storing data owner's source file.

This work was supported by the National Natural Science Foundation of China under Grant 62072078, the Hunan Provincial Natural Science Foundation of China under Grant 2018JJ3191. *(Corresponding author: Xiong Li.)*

X. Li is with the Institute for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China and with the Faculty of Computer Science, University of New Brunswick, Fredericton, E3B 5A3, Canada (e-mail: lixiongzhq@163.com).

S. Liu is with the School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, 411201, China (e-mail: liushanpeng0@gmail.com).

R. Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, E3B 5A3, Canada (e-mail: rlu1@unb.ca).

X. Zhang is with the Institute for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China and with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, Guangdong 518040, China (email: johnsonzxs@uestc.edu.cn).

II. Review of Shang *et al.*'s protocol

In this section, we briefly review Shang *et al.*'s identity-based dynamic data auditing protocol for big data storage [3], which mainly includes four entities, *i.e.*, KGC (Key Generate Center), DO (data owner), SP (service provider) and TPA (third-party auditor), where KGC is responsible for the parameter generation and SP is an untrusted party. Specifically, their protocol comprises eight algorithms, *i.e.*, Setup, Extract, TagGen, Challenge, GenProof, CheckProof, ExcuteUpdate and VerifyUpdate. For the eight algorithms, we will omit reviewing the last two dynamic operation algorithms in their protocol, as they are not directly related to our attack, and the detailed information about that part can be found in [3]. In the following, we review the first six algorithms of Shang *et al.*'s protocol in detail.

Setup: KGC chooses two multiplicative cyclic groups G_1 and G_2 with same prime order q, and g is a generator of G_1. KGC defines an efficiently computable bilinear pairing $e : G_1 \times G_1 \rightarrow G_2$. Then, KGC generates a random element $msk \in Z_q$ as a master secret key and computes the corresponding master public key $P_{pub} = g^{msk}$. Next, four hash functions, $H : \{0,1\}^* \rightarrow \{0,1\}^l$, $H_1, H_2 : \{0,1\}^* \rightarrow G_1$, and $H_3 : G_2 \rightarrow \{0,1\}^l (l \in Z_q)$ are generated by KGC. Finally, KGC generates a random verification code $v_0 \in Z_q$, and distributes it to DO and SP. In addition, according to Shang *et al.*'s protocol, KGC will update a new v_0 for DO and SP after each auditing process.

Extract: Based on DO's identity $ID \in \{0,1\}^*$, KGC extracts DO's private key $sk = H_1(ID)^{msk} \in G_1$ by using the master secret key msk.

TagGen: For a file F with name $fname$, DO first divides it into n blocks $F = \{m_1, m_2, \cdots, m_n\}$. DO picks a random number $\eta \in Z_q$ and computes $r = g^{\eta} \in G_1$. For each block m_i $(i = \{1, 2, \cdots, n\})$, DO computes a tag $\sigma_i = sk^{m_i} \cdot H_2(fname)^{\eta}$, and all blocks form a tag set $\Phi = \{\sigma_1, \sigma_2, \cdots, \sigma_n\}$. In addition, DO computes $H_0 = H(m_1 \| m_2 \| \cdots \| m_n \| v_0)$. Then, DO generates a root R based on the construction of Merkle hash tree, where the leave nodes of the tree form an ordered set of hashes of tags. Next, DO signs the root R with sk as a signature $sig_{sk}(R)$. Finally, DO respetively sends $\{F, \Phi, sig_{sk}(R)\}$ to SP, and $H_0 = H(m_1 \| m_2 \| \cdots \| m_n \| v_0)$ to TPA, and deletes the local source file F.

Challenge: TPA first generates a set with n random elements from Z_q, *i.e.*, $Q = \{v_1, v_2, \cdots, v_n\}$. Then, TPA chooses a random number $\rho \in Z_q$, and computes $c_1 = g^{\rho}$, $Z = e(H_1(ID), P_{pub})$ and $c_2 = Z^{\rho}$, and sends a challenge $chal = (c_1, c_2, Q)$ to SP.

GenProof: Upon receiving the challenge from TPA, SP first computes $H_0' = H(m_1 \| m_2 \| \cdots \| m_n \| v_0)$, $\mu = \sum_{i \in I} v_i m_i$, $\sigma = \prod_{i \in I} \sigma_i^{v_i}$, where I is the index set. Then, SP computes $m' = H_3(e(\sigma, c_1) \cdot c_2^{-\mu})$, and responds the proof (m', r, H_0') to TPA.

CheckProof: Once getting the proof, TPA first checks $H_0 \overset{?}{=} H_0'$. TPA outputs FALSE if it is not held. Otherwise, TPA continues to check $m' \overset{?}{=} H_3(\prod_{i\in I} e(H_2(fname)^{v_i}, r^{\rho}))$. TPA outputs TRUE if they are equal, and the FALSE will be output otherwise.

III. Security analysis of Shang *et al.*'s protocol

As we know, the security of the whole system is highly dependent on the security of private keys, and the leakage of a private key will greatly damage the security of the system. Unfortunately, Shang *et al.*'s protocol [3] is vulnerable to a private key reveal attack, and SP can reveal DO's private key via the outsourced data. Subsequently, SP can generate a valid proof to pass the challenge of TPA even if all block and tag pairs of DO have been deleted by SP. In this section, we describe this attack on Shang *et al.*'s protocol [3] in detail as follows.

As shown in the description of **TagGen** algorithm, DO's data $(m_i, \sigma_i = sk^{m_i} \cdot \underline{H_2(fname)^{\eta}})(i = \{1, 2, \cdots, n\})$ will be stored in SP. Therefore, upon receiving these data, SP computes $H_0' = H(m_1\|m_2\|\cdots\|m_n\|v_0)$ and stores them in the database. Here, we should be aware that the component $H_2(fname)^{\eta}$ is constant for all tags $\sigma_i, i = \{1, 2, \cdots, n\}$. Therefore, SP can reveal DO's private key sk by the following steps:

Step 1. The SP picks up two data pairs from the stored file, for example (m_j, σ_j) and (m_k, σ_k), where $j, k \in \{1, 2, \cdots, n\}$.

Step 2. The SP computes

$$\frac{\sigma_j}{\sigma_k} = \frac{sk^{m_j} \cdot H_2(fname)^{\eta}}{sk^{m_k} \cdot H_2(fname)^{\eta}} = \frac{sk^{m_j}}{sk^{m_k}} = sk^{m_j - m_k}$$

Step 3. With the source blocks m_j and m_k, SP computes $(m_j - m_k)^{-1}$ mod q, and then DO's private key sk can be revealed by SP as

$$(\frac{\sigma_j}{\sigma_k})^{(m_j - m_k)^{-1}} = sk^{(m_j - m_k)\cdot(m_j - m_k)^{-1}} = sk$$

Step 4. After revealing DO's private key sk, SP can obtain $H_2(fname)^{\eta}$ by calculating

$$\frac{\sigma_j}{sk^{m_j}} = \frac{sk^{m_j} \cdot H_2(fname)^{\eta}}{sk^{m_j}} = H_2(fname)^{\eta}$$

With DO's private key sk and $H_2(fname)^{\eta}$, SP can generate a valid proof for a challenge even if all $(m_i, \sigma_i)(i = \{1, 2, \cdots, n\})$ pairs have been deleted by SP. We illustrate the process as below:

Step 1. TPA generates a set $Q = \{v_1, v_2, \cdots, v_n\}$ with n random elements from Z_q. Then, TPA chooses a random number $\rho \in Z_q$, and computes $c_1 = g^{\rho}$, $Z = e(H_1(ID), P_{pub})$ and $c_2 = Z^{\rho}$, and sends a challenge $chal = (c_1, c_2, Q)$ to SP.

Step 2. Upon receiving the challenge from TPA, SP computes

$$\mu = \sum_{i\in I} v_i, \sigma = \prod_{i\in I}(sk \cdot H_2(fname)^{\eta})^{v_i},$$

where I is the index set. Then, SP computes

$$m' = H_3(e(\sigma, c_1) \cdot c_2^{-\mu}),$$

and responds the proof (m', r, H_0') to TPA.

Step 3. TPA affirms that

$$H_0 = H_0' \text{ and } m' = H_3(\prod_{i\in I} e(H_2(fname)^{v_i}, r^{\rho})),$$

and TPA believes that the data stored on the SP is complete.

Here, we should note that the **Setup** algorithm of Shang *et al.*'s protocol [3] requires the KGC to update a new v_0^{new} for DO and SP after each auditing process. The original intention is to update a new $H_0 = H_2(m_1\|m_2\|\cdots\|m_n\|v_0^{new})$ between DO and TPA. However, DO's source file $F = \{m_1, m_2, \cdots, m_n\}$ has been deleted locally and thus actually DO cannot compute a new $H_0 = H_2(m_1\|m_2\|\cdots\|m_n\|v_0^{new})$ without F. Therefore, $H_0' = H_2(m_1\|m_2\|\cdots\|m_n\|v_0^{new})$ can not be updated in their protocol, SP just needs to store H_0 in the database, and then TPA will see the equation $H_0 = H_0'$ holds on every verification. Besides it, we should note that we can use $\sigma = \prod_{i\in I}(sk \cdot H_2(fname)^{\eta})^{v_i} = H_1(ID)^{msk\cdot\sum_{i\in I} v_i} \cdot (\prod_{i\in I} H_2(fname)^{v_i})^{\eta}$ and $c_2^{-\mu}$ to eliminate the comment parts and just leave $\prod_{i\in I} H_2(fname)^{v_i})^{\eta}$ in m', and the detail explanation of the equation $m' = H_3(\prod_{i\in I} e(H_2(fname)^{v_i}, r^{\rho}))$ holds is as follows:

$$\begin{aligned}
& m' \\
&= H_3(e(\sigma, c_1) \cdot c_2^{-\mu}) \\
&= H_3(e(\sigma, g^{\rho}) \cdot e(H_1(ID), P_{pub})^{-\rho\cdot\mu}) \\
&= H_3\left(\frac{e(\prod_{i\in I}(sk \cdot H_2(fname)^{\eta})^{v_i}, g^{\rho})}{e(H_1(ID), g^{msk})^{\rho\cdot\mu}}\right) \\
&= H_3\left(\frac{e(\prod_{i\in I} sk^{v_i} \cdot (\prod_{i\in I} H_2(fname)^{\eta})^{v_i}, g^{\rho})}{e(H_1(ID)^{msk\cdot\mu}, g^{\rho})}\right) \\
&= H_3\left(\frac{e(\prod_{i\in I} H_1(ID)^{msk\cdot v_i} \cdot (\prod_{i\in I} H_2(fname)^{v_i})^{\eta}, g^{\rho})}{e(H_1(ID)^{msk\cdot\sum_{i\in I} v_i}, g^{\rho})}\right) \\
&= H_3\left(\frac{e(H_1(ID)^{msk\cdot\sum_{i\in I} v_i} \cdot (\prod_{i\in I} H_2(fname)^{v_i})^{\eta}, g^{\rho})}{e(H_1(ID)^{msk\cdot\sum_{i\in I} v_i}, g^{\rho})}\right) \\
&= H_3\left(e\left(\frac{H_1(ID)^{msk\cdot\sum_{i\in I} v_i} \cdot (\prod_{i\in I} H_2(fname)^{v_i})^{\eta}}{H_1(ID)^{msk\cdot\sum_{i\in I} v_i}}, g^{\rho}\right)\right) \\
&= H_3\left(e\left(\prod_{i\in I}(H_2(fname)^{v_i})^{\eta}, g^{\rho}\right)\right) \\
&= H_3\left(e\left(\prod_{i\in I} H_2(fname)^{v_i}, g^{\eta\cdot\rho}\right)\right) \\
&= H_3\left(\prod_{i\in I} e(H_2(fname)^{v_i}, r^{\rho})\right)
\end{aligned}$$

From the above analysis, we can see that SP can reveal DO's private key sk, and then generate a valid proof to pass the challenge of TPA even if all data pairs $(m_i, \sigma_i)(i = \{1, 2, \cdots, n\})$ have been deleted. The reason why Shang *et al.*'s scheme [3] is vulnerable to this private key reveal attack is that the value $H_2(fname)^{\eta}$ in each tag $\sigma_i = sk^{m_i} \cdot \underline{H_2(fname)^{\eta}}(i = \{1, 2, \cdots, n\})$ is fixed and the same. As a result, SP can easily recover DO's private key sk and generate a valid proof without using DO's data blocks and tags to pass the challenge of TPA. Like most of work in this field, this attack can be avoided by including the block number indicator in the tags, such as let $\sigma_i = sk^{m_i} \cdot \underline{H_2(fname\|i)^{\eta}}(i = \{1, 2, \cdots, n\})$, and adjust the proof generation and verification algorithms accordingly.

IV. Conclusion

The security of the private keys is the cornerstone of all security protocols to achieve other security properties, and the private keys of all system participants should not be revealed by any malicious entity. In this paper, we pointed out a security weakness of an identity-based data integrity auditing protocol for big data storage proposed by Shang *et al.* [3]. Concretely, their protocol is vulnerable to a secret key reveal attack that SP can calculate DO's private key sk from the outsourced data. Then, SP can generate a valid proof to pass the challenge of TPA even if it has deleted all data pairs $(m_i, \sigma_i)(i = \{1, 2, \cdots, n\})$ of DO. We hope that by identifying this flaw, similar weaknesses can be avoided in future designs for identity-based integrity auditing protocols. Although there are many schemes have been proposed in the field of public auditing for cloud computing, there is room for further research in this field. On the one hand, it is a problem worthy of study that how to minimize the DO's computational cost while ensuring the functions and security features. On the other hand, we can target to design a secure identity-based public auditing scheme for cloud storage that supports batch verification and dynamic operations.

References

[1] Y. Wang, Q. Wu, B. Qin, W. Shi, R. H. Deng, and J. Hu, "Identity-based data outsourcing with comprehensive auditing in clouds," *IEEE transactions on information forensics and security*, vol. 12, no. 4, pp. 940–952, 2017.

[2] Y. Yu, M. H. Au, G. Ateniese, X. Huang, W. Susilo, Y. Dai, and G. Min, "Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 767–778, 2017.

[3] T. Shang, F. Zhang, X. Chen, J. Liu, and X. Lu, "Identity-based dynamic data auditing for big data storage," *IEEE Transactions on Big Data*, doi: 10.1109/TBDATA.2019.2941882, 2019.