Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 8062182, 6 pages
https://doi.org/10.1155/2017/8062182

Research Article

Attack on Privacy-Preserving Public Auditing Schemes for Cloud Storage

Baoyuan Kang, Jiaqiang Wang, and Dongyang Shao

School of Computer Science and Software, Tianjin Polytechnic University, Tianjin 300387, China

Correspondence should be addressed to Baoyuan Kang; baoyuankang@aliyun.com

Received 9 December 2016; Accepted 19 April 2017; Published 11 May 2017

Academic Editor: Emilio Insfran

With the development of Internet, cloud computing has emerged to provide service to data users. But, it is necessary for an auditor on behalf of users to check the integrity of the data stored in the cloud. The cloud server also must ensure the privacy of the data. In a usual public integrity check scheme, the linear combination of data blocks is needed for verification. But, after times of auditing on the same data blocks, based on collected linear combinations, the auditor might derive these blocks. Recently, a number of public auditing schemes with privacy-preserving are proposed. With blinded linear combinations of data blocks, the authors of these schemes believed that the auditor cannot derive any information about the data blocks and claimed that their schemes are provably secure in the random oracle model. In this paper, with detailed security analysis of these schemes, we show that these schemes are vulnerable to an attack from the malicious cloud server who modifies the data blocks and succeeds in forging proof information for data integrity check.

1. Introduction

With the development of Internet, cloud computing has emerged. Cloud computing is a new model of computing in contrast to conventional computing. This new paradigm allows data users to outsource their data to a cloud service provider. The term cloud refers to a thousand of virtualized servers distributed over a set of data centers with different geographical locations connected together through telecommunication links [1]. The services on the cloud are delivered to the users as pay-as-you-go pricing model.

Although cloud computing offers various advantages to both users and the cloud service provider, and is envisioned as a promising service platform for the next generation Internet, security and privacy are the major challenges which inhibit the cloud computing wide acceptance in practice. Once data users transfer their data to the cloud, users lose their physical control over data. The outsourced data on the cloud are at risk from internal and external threats. The first threat is that the cloud service provider might delete less frequently accessed data. So, users need to make sure their data remain intact after uploading to the cloud, and data integrity check is becoming vital. As data users no longer physically possess the storage of their data and are confined by resource capability, traditional integrity checking technologies are not well suited for the cloud environment. Data users hope one-third party on their behalf to verify their data integrity. The issue of public auditing for data integrity check is proposed.

After Ateniese et al.'s first work [2], people proposed many public auditing schemes [3–16] for data integrity check. In a typical public auditing scheme, there are three characters, one data user, one cloud server, and one auditor. The data user transfers his data to the cloud for storage and computing. On behalf of the user the auditor, who has experience and capability, is responsible for the data integrity check. Before sending data to the cloud, the user divides a data file into many data blocks. Then, using signature technology the user generates an authentication tag for each block. These tags are sent to the cloud server with data blocks. To check the integrity of the outsourced data file, using sampling test idea, the auditor sends challenging information to the cloud server. Upon receiving the challenging information the cloud server generates a response by the data blocks and corresponding block tags and sends the response to the auditor. Then, the auditor verifies the validity of the response. If the response is valid, the

auditor and the user believe the outsourced data file remain intact.

In the security model of public auditing schemes, the user is honest. But the cloud server is a semitrusted party. As mentioned earlier, the cloud server might delete less frequently accessed data for his benefit. The auditor is honest but curious. The auditor might obtain some information of the data in auditing process. So, secure public auditing scheme should also satisfy the privacy-preserving requirement. In fact, in many existing schemes, the linear combinations of data blocks are needed for verification without data privacy guarantee against the auditor. The users, who rely on the auditor just for the storage security of their data, do not want the auditing process leaking any information of their data. But, based on collected linear combinations of the same data blocks in times of check, the auditor might derive these data blocks.

Recently, some public auditing schemes [17–21] concerning privacy-preserving are proposed. In [21], Li et al. proposed a privacy-preserving cloud data auditing scheme with efficient key update and claimed their scheme is proved secure in the random oracle model. The difference between Li et al.'s scheme and other existing schemes is that in Li et al.'s scheme each block is further fragmented into a certain number of sectors, and the authenticator for each block is related to its each sector. In [19], Wang et al. proposed a privacy-preserving public auditing scheme for secure cloud storage and claimed that their scheme is provably secure and highly efficient. In [17], Wang et al. proposed a privacy-preserving public auditing scheme. But, in [18] Worku et al. showed that in Wang et al.'s scheme [17] the malicious cloud server can forge a signature for his any selected block. So, once the server possesses data from users, he can modify the data as he wants. Worku et al. also proposed an efficient privacy-preserving public auditing scheme and claimed that the proposed scheme is proved secure in the random oracle model. However, in this paper, we will point that these schemes [18, 19, 21] are insecure. The malicious cloud server against these schemes can break the data integrity without being found by the auditor.

The rest of the paper is organized as follows. In Section 2, we review bilinear pairing and computational Diffie-Hellman problem relevant to the security of the discussed schemes. In Section 3, we review Li et al.'s scheme. We show an attack on Li et al.'s scheme in Section 4. In Section 5, we review Worku et al.'s scheme. We demonstrate that Worku et al.'s scheme and Wang et al.'s scheme are subjected to the same attack In Sections 6 and 7, respectively. Conclusion is given in Section 8.

2. Preliminary

2.1. The Bilinear Pairing. Let G_1 be a cyclic additive group generated by P, whose order is a prime q, and G_2 be a cyclic multiplicative group of the same order. Let $e : G_1 \times G_1 \to G_2$ be a pairing map which satisfies the following conditions:

(1) Bilinearity: for any $P, Q, R \in G_1$, then

$$e(P+Q,R) = e(P,R)\,e(Q,R),$$
$$e(P,Q+R) = e(P,Q)\,e(P,R). \tag{1}$$

In particular, for any $a, b \in Z_q$, $e(aP, bP) = e(P, abP) = e(abP, P) = e(P,P)^{ab}$.

(2) Nondegeneracy: there exists $P, Q \in G_1$, such that $e(P,Q) \neq 1$.

(3) Computability: there is an efficient algorithm to compute $e(P,Q)$ for all $P, Q \in G_1$.

2.2. Computational Diffie-Hellman (CDH) Problem. Given a generator P of an additive cyclic group G with order q and given (aP, bP) for unknown $a, b \in Z_q^*$, one computes abP.

3. Brief Review of Li et al.'s Scheme

In [21], Li et al. proposed a privacy-preserving cloud data auditing scheme with key update. Here we review it but omit the content related to key update.

CrsGen. On input of a security parameter λ, this algorithm outputs a large prime p and G, G_T, two multiplicative cyclic groups of the same order p. g is a generator of G. $e : G \times G \to G_T$ denotes a bilinear map and $H_0, H_1 : \{0,1\}^* \to G$ represent two collision resistant cryptographic hash functions. In addition, this algorithm picks randomly $h, u_1, u_2, \ldots, u_s \in G$ and computes $\eta = e(g,h)$. The common reference string crs is $(p, G, G_T, g, e, H_0, H_1, h, u_1, u_2, \ldots, u_s, \eta)$.

KeyGen. On input of the common reference string crs, a cloud user generates a signing key pair (spk, ssk), $\text{spk} = g^{\text{ssk}}$, and another key pair (a, v) for generating authenticators of file blocks, where $a \in Z_p$ and $v = g^a$. The secret key of the data user is $\text{sk} = (a, \text{ssk})$ and the public key is $\text{pk} = (\text{spk}, v)$. For convenience, Let $\eta_i = e(u_i, v)$, $i = 1, \ldots, s$.

AuthGen. Given a file F, the data owner firstly applies erasure codes such as RS code to obtain a processed file F' and splits F' into n blocks. Each block is further fragmented into s sectors $\{m_{ij}\}_{1 \le i \le n, 1 \le j \le s}$, which is an element of Z_p. The data user selects a file name Fn from a sufficiently large domain. Let $t_0 = \text{Fn} \parallel n$. The data user computes $t = (H_0(t_0))^{\text{ssk}_1}$ and denotes the file tag $\text{ft} = t_0 \parallel t$. Then, for each i, $1 \le i \le n$, the user computes an authenticator σ_i for block i as

$$\sigma_i = \left(H_1(\text{Fn} \parallel i) \cdot \prod_{j=1}^{s} u_j^{m_{ij}} \right)^a. \tag{2}$$

Finally, the data owner stores

$$\text{ft} \parallel \{m_{ij}\}_{1 \le i \le n, 1 \le j \le s} \parallel \text{Metadata} \tag{3}$$

to the cloud, where $\text{Metadata} = \{\sigma_i\}_{1 \le i \le n}$.

Proof. This is a 5-move interactive proof protocol executed between the cloud server and the auditor (TPA) as follows.

(1) The TPA picks a random integer c and $k, \varphi \in Z_p$, computing $\psi = g^k h^\varphi$. For $1 \le i \le c$, the TPA selects a random $v_i \in Z_p$. The commitment ψ and the challenge chal $= \{i, v_I\}_{1 \le i \le c}$, which locates the positions of the challenged blocks in this auditing process, are sent to the cloud server.

(2) Upon receiving (chal, ψ), the cloud server firstly chooses $r, \rho_r, \rho_1, \ldots, \rho_s \in Z_p$ randomly and then computes

$$\begin{aligned} \omega &= \prod_{(i,v_i)\in \text{chal}} \sigma_i^{v_i} \cdot h^r, \\ T &= \eta^{\rho_r} \eta_1^{\rho_1} \cdots \eta_s^{\rho_s} \end{aligned} \tag{4}$$

and forwards (T, ω) to the TPA.

(3) The TPA sends (k, φ) to the server.

(4) The server checks if $\psi = g^k h^\varphi$. If the equation does not hold, the server aborts. Otherwise, he computes

$$\begin{aligned} z_r &= \rho_r - kr, \\ \mu_j &= \sum_{(i,v_i)\in \text{chal}} v_i m_{ij}, \\ z_j &= \rho_j - k\mu_j \\ &\qquad (1 \le j \le s) \end{aligned} \tag{5}$$

and sends $(z_r, z_1, \ldots, z_s)$ to the TPA.

(5) The TPA verifies the file tag ft firstly by checking if the following equation holds:

$$e(g, t) = e(\text{spk}, H_0(t_0)). \tag{6}$$

Then, TPA verifies the equation

$$\left(\frac{e(\omega, g)}{e\left(\prod_{(i,v_i)\in \text{chal}} H_1(\text{Fn} \parallel i)^{v_i}, v\right)} \right)^k = \frac{T}{\eta^{z_r} \eta_1^{z_1} \cdots \eta_s^{z_s}}. \tag{7}$$

□

4. Attack on Li et al.'s Scheme

In this section, we show that Li et al.'s scheme is vulnerable to a modifying attack on data integrity check.

In proof phase, the malicious cloud server can change data blocks by modifying blocks sectors. He changes

$$\begin{aligned} \mu_j &= \sum_{(i,v_i)\in \text{chal}} v_i m_{ij}, \\ z_j &= \rho_j - k\mu_j \end{aligned} \tag{8}$$

into

$$\begin{aligned} \overline{\mu}_j &= \sum_{(i,v_i)\in \text{chal}} v_i \left(b m_{ij}\right), \\ \overline{z}_j &= \rho_j - k b^{-1} \overline{\mu}_j, \end{aligned} \tag{9}$$

respectively, where $b \in Z_p$ is randomly selected by the server. Other computations remain unchanged. Now, the forged proof information

$$(T, \omega, z_r, \overline{z}_1, \ldots, \overline{z}_s) \tag{10}$$

can pass the author's verification.

Theorem 1. *The forged proof information $(T, \omega, z_r, \overline{z}_1, \ldots, \overline{z}_s)$ produced in the above analysis can pass the auditor's verification.*

Proof. In fact,

$$\begin{aligned} &\left(\frac{e(\omega, g)}{e\left(\prod_{(i,v_i)\in \text{chal}} H_1(\text{Fn} \parallel i)^{v_i}, v\right)} \right)^k \\ &= \left(\frac{e\left(\prod_{(i,v_i)\in \text{chal}} \sigma_i^{v_i} \cdot h^r, g\right)}{e\left(\prod_{(i,v_i)\in \text{chal}} H_1(\text{Fn} \parallel i)^{v_i}, v\right)} \right)^k \\ &= \left(\frac{e\left(\prod_{(i,v_i)\in \text{chal}} \left(H_1(\text{Fn} \parallel i) \cdot \prod_{j=1}^s u_j^{m_{ij}}\right)^{av_i} \cdot h^r, g\right)}{e\left(\prod_{(i,v_i)\in \text{chal}} H_1(\text{Fn} \parallel i)^{v_i}, v\right)} \right)^k \\ &= \left(\frac{e\left(\prod_{(i,v_i)\in \text{chal}} \left(H_1(\text{Fn} \parallel i) \cdot \prod_{j=1}^s u_j^{m_{ij}}\right)^{v_i}, g^a\right) \cdot e(h^r, g)}{e\left(\prod_{(i,v_i)\in \text{chal}} H_1(\text{Fn} \parallel i)^{v_i}, v\right)} \right)^k \\ &= \left(\frac{e\left(\prod_{(i,v_i)\in \text{chal}} \left(H_1(\text{Fn} \parallel i) \cdot \prod_{j=1}^s u_j^{m_{ij}}\right)^{v_i}, v\right) \cdot e(h^r, g)}{e\left(\prod_{(i,v_i)\in \text{chal}} H_1(\text{Fn} \parallel i)^{v_i}, v\right)} \right)^k \\ &= \left(e\left(\prod_{(i,v_i)\in \text{chal}} \left(\prod_{j=1}^s u_j^{m_{ij}} \right)^{v_i}, v \right) \cdot e(h^r, g) \right)^k. \end{aligned} \tag{11}$$

But,

$$\begin{aligned} \frac{T}{\eta^{z_r} \eta_1^{\overline{z}_1} \cdots \eta_s^{\overline{z}_s}} &= \frac{\eta^{\rho_r} \eta_1^{\rho_1} \cdots \eta_s^{\rho_s}}{\eta^{z_r} \eta_1^{\overline{z}_1} \cdots \eta_s^{\overline{z}_s}} = \eta^{\rho_r - z_r} \eta_1^{\rho_1 - \overline{z}_1} \cdots \eta_s^{\rho_s - \overline{z}_s} \\ &= \eta^{kr} \eta_1^{kb^{-1}\overline{\mu}_1} \cdots \eta_s^{kb^{-1}\overline{\mu}_s} = \left(\eta^r \eta_1^{b^{-1}\overline{\mu}_1} \cdots \eta_s^{b^{-1}\overline{\mu}_s} \right)^k \\ &= \Big(e(g, h)^r \cdot e(u_1, v)^{b^{-1} \sum_{(i,v_i)\in \text{chal}} v_i (b m_{i1})} \\ &\quad \cdots e(u_s, v)^{b^{-1} \sum_{(i,v_i)\in \text{chal}} v_i (b m_{is})} \Big)^k = \Big(e(h^r, g) \\ &\quad \cdot e\left(u_1^{\sum_{(i,v_i)\in \text{chal}} v_i m_{i1}} \cdots u_s^{\sum_{(i,v_i)\in \text{chal}} v_i m_{is}}, v\right) \Big)^k \\ &= \left(e\left(\prod_{(i,v_i)\in \text{chal}} \left(\prod_{j=1}^s u_j^{m_{ij}} \right)^{v_i}, v \right) \cdot e(h^r, g) \right)^k. \end{aligned} \tag{12}$$

So,

$$\left(\frac{e(\omega, g)}{e\left(\prod_{(i,v_i)\in \text{chal}} H_1(\text{Fn} \parallel i)^{v_i}, v\right)} \right)^k = \frac{T}{\eta^{z_r} \eta_1^{\overline{z}_1} \cdots \eta_s^{\overline{z}_s}}. \tag{13}$$

$(T, \omega, z_r, \overline{z}_1, \ldots, \overline{z}_s)$ passes the auditor's verification; it is valid proof information. The malicious cloud server succeeds in modifying attack on data integrity check. □

5. Brief Review of Worku et al.'s Scheme

In this section, we give a brief review of Worku et al.'s scheme [18], which is composed of four algorithms.

Let $G_1 = G_2 = G$ and $e : G \times G \to G_T$ be a bilinear map, where G and G_T are multiplicative cyclic groups of prime

order p. Let g be a generator of G. Let $H : \{0,1\}^* \to G$ be a hash function, which maps strings to G, and let $h(\cdot) : G \to Z_p$ be another hash function which maps group of elements of G uniformly to Z_p.

KeyGen. The data user first generates a random signing key pair (ssk, spk) and then chooses $x \leftarrow {}^R Z_p$ and $u \leftarrow {}^R G$ and computes $v = g^x$. The user then states sk = $(x$, ssk) as his/her secret key and pk = $(u, v, g,$ spk) as public parameters.

SigGen. For file naming, the user chooses a random element name in Z_p for file $F = \{m_i\}_{1 \le i \le n}$ and computes the file tag as $t =$ name $\| \operatorname{Sig}_{\text{ssk}}$(name). Next, for each block $m_i \in Z_p$, user generates a signature σ_i as follows:

$$\sigma_i = \left(H(i) \cdot u^{m_i}\right)^x. \tag{14}$$

Then, finally, the user sends $\{F, \phi = \{\sigma_i\}_{1 \le i \le n}, t\}$ to the cloud server for storage and deletes the file and its corresponding set of signatures from local storage. Any time when the auditor wants to start the auditing protocol, first he retrieves the file tag t for F and checks its validity using spk and quits if failed. If the proof on t is correct, the auditor sends a challenge chal to the server. That is, the auditor picks random elements c, k_1, k_2 in Z_p and sends chal $= (c, k_1, k_2)$ to the server where k_1 and k_2 are pseudorandom permutation keys chosen randomly by the auditor for each auditing.

ProofGen. After receiving the challenge, the server first determines the subset $I = \{s_j\}$ $(1 \le j \le c)$ of set $[1, n]$ using pseudorandom permutation $\pi_{\text{key}}(\cdot)$ as $s_j = \pi_{k_1}(j)$ and it also determines $v_{s_j} = f_{k_2}(j)$ $(1 \le j \le c)$ using pseudorandom function $f_{\text{key}}(\cdot)$. Finally, for $i \in I$, server computes

$$\begin{aligned} \mu^* &= \sum_{i=s_1}^{s_c} v_i m_i, \\ \sigma &= \prod_{i=s_1}^{s_c} \sigma_i^{v_i}. \end{aligned} \tag{15}$$

For blinding, the server chooses a random element $r \leftarrow Z_p$, using the same pseudorandom function, as $r = f_{k_3}$(chal), where k_3 is a pseudorandom function key generated by the server for each auditing. The server then calculates $R = u^r$ and computes $\mu = \mu^* + rh(R)$ and, then, sends (μ, σ, R) to the auditor.

VerifyProof. Upon receiving the proof (μ, σ, R) TPA computes $s_j = \pi_{k_1}(j)$ and $v_{s_j} = f_{k_2}(j)$ $(1 \le j \le c)$, where $1 \le j \le c$. Finally, the auditor verifies the proof by checking the following equation and outputs "True" if valid and "False" otherwise:

$$e(\sigma, g) = e\left(\prod_{i \in I} H(i)^{v_i} \cdot u^{\mu} \cdot R^{-h(R)}, v\right). \tag{16}$$

6. Attack on Worku et al.'s Scheme

In this section, we demonstrate that the malicious cloud server can break the integrity check by modification attack.

Suppose a file M from the data user is divided into n blocks; that is, $= m_1 \| m_2 \| \cdots \| m_n$. Let σ_i be m_i's authentication tag. Let A be a malicious cloud server. When A receives the file M, A might replace each file block m_i with $a \cdot m_i$. Here $a (\in Z_p)$ is randomly selected by A. Upon receiving the challenge information, in ProofGen phase, A can change

$$\begin{aligned} \mu^* &= \sum_{i=s_1}^{s_c} v_i m_i, \\ \mu &= \mu^* + rh(R) \end{aligned} \tag{17}$$

into

$$\begin{aligned} \overline{\mu}^* &= \sum_{i=s_1}^{s_c} v_i (a m_i), \\ \overline{\mu} &= a^{-1} \cdot \overline{\mu}^* + rh(R), \end{aligned} \tag{18}$$

respectively. Other computations remain unchanged. Then, the forged proof information

$$(\overline{\mu}, \sigma, R) \tag{19}$$

can pass the author's verification.

Theorem 2. *The forged proof information $(\overline{\mu}, \sigma, R)$ produced in the above analysis can pass the auditor's verification*

Proof. In fact, based on the equations

$$\begin{aligned} \overline{\mu}^* &= \sum_{i=s_1}^{s_c} v_i (a m_i), \\ \overline{\mu} &= a^{-1} \cdot \overline{\mu}^* + rh(R) \end{aligned} \tag{20}$$

produced by the malicious cloud server, the following derivation is established:

$$\begin{aligned} e(\sigma, g) &= e\left(\prod_{i=s_1}^{s_c} \sigma_i^{v_i}, g\right) = e\left(\prod_{i=s_1}^{s_c} \left(H(i) \cdot u^{m_i}\right)^{x v_i}, g\right) \\ &= e\left(\prod_{i=s_1}^{s_c} H(i)^{v_i} \cdot u^{\sum_{i=s_1}^{s_c} m_i v_i}, g^x\right) \\ &= e\left(\prod_{i=s_1}^{s_c} H(i)^{v_i} \cdot u^{a^{-1}\overline{\mu}^*}, v\right) \\ &= e\left(\prod_{i=s_1}^{s_c} H(i)^{v_i} \cdot u^{\overline{\mu} - rh(R)}, v\right) \\ &= e\left(\prod_{i=s_1}^{s_c} H(i)^{v_i} \cdot u^{\overline{\mu}} \cdot R^{-h(R)}, v\right). \end{aligned} \tag{21}$$

So, $(\overline{\mu}, \sigma, R)$ passes the auditor's verification, and it is valid proof information. The malicious cloud server that modifies the file blocks succeeds in deceiving the auditor. □

7. Attack on Wang et al.'s Scheme

To save space we do not review Wang et al.'s scheme. For its detailed description, readers can refer to literature [19]. Due to similarity, Wang et al.'s scheme is subjected to the above attack.

When the malicious cloud server A receives a data file $M = m_1 \parallel m_2 \parallel \cdots \parallel m_n$, similarly, A might replace each file block m_i with $a \cdot m_i$. Here $a(\in Z_p)$ is selected by A. Upon receiving the challenge information, in ProofGen phase malicious cloud server A can change

$$\begin{aligned} \mu' &= \sum_{i=s_1}^{s_c} v_i m_i, \\ \mu &= r + \gamma \mu' \end{aligned} \tag{22}$$

into

$$\begin{aligned} \overline{\mu}' &= \sum_{i=s_1}^{s_c} v_i (a m_i), \\ \overline{\mu} &= r + a^{-1} \cdot \gamma \cdot \overline{\mu}', \end{aligned} \tag{23}$$

respectively. Other computations remain unchanged. Then, the forged proof information

$$(\overline{\mu}, \sigma, R) \tag{24}$$

can pass the author's verification.

Theorem 3. *The forged proof information $(\overline{\mu}, \sigma, R)$ produced in the above analysis can pass the auditor's verification*

Proof. In fact, due to the equations

$$\begin{aligned} \overline{\mu}' &= \sum_{i=s_1}^{s_c} v_i (a m_i), \\ \overline{\mu} &= r + a^{-1} \cdot \gamma \cdot \overline{\mu}' \end{aligned} \tag{25}$$

produced by the malicious cloud server, the following derivation is established:

$$\begin{aligned} R \cdot e(\sigma^\gamma, g) &= e(u, v)^r \\ &\quad \cdot e\left(\left(\prod_{i=s_1}^{s_c} (H(W_i) \cdot u^{m_i})^{x v_i}\right)^\gamma, g\right) \\ &= e(u^r, v) \\ &\quad \cdot e\left(\prod_{i=s_1}^{s_c} (H(W_i)^{v_i} \cdot u^{v_i m_i})^\gamma, g\right)^x \\ &= e(u^r, v) \\ &\quad \cdot e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^\gamma \cdot u^{\sum_{i=s_1}^{s_c} v_i m_i \gamma}, g^x\right) \\ &= e(u^r, v) \\ &\quad \cdot e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^\gamma \cdot u^{a^{-1}\overline{\mu}'\gamma}, v\right) \\ &= e(u^r, v) \\ &\quad \cdot e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^\gamma \cdot u^{\overline{\mu} - r}, v\right) \\ &= e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^\gamma \cdot u^{\overline{\mu}}, v\right). \end{aligned} \tag{26}$$

So, $(\overline{\mu}, \sigma, R)$ passes the auditor's verification, it is valid proof information. The malicious cloud server succeeds in deceiving the auditor. □

8. Conclusion

In this paper, we analyze three existing privacy-preserving public auditing schemes for secure cloud storage. We demonstrate an attack against them. In the attack, the malicious cloud server that modifies the data blocks succeeds in forging proof information for data integrity check. As far as we know, it is an open problem to propose secure privacy-preserving public auditing schemes.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the Applied Basic and Advanced Technology Research Programs of Tianjin (no. 15JCYBJC15900).

References

[1] M. Sookhak, H. Talebian, E. Ahmed, A. Gani, and M. K. Khan, "A review on remote data auditing in single cloud server: taxonomy and open issues," *Journal of Network and Computer Applications*, vol. 43, pp. 121–141, 2014.

[2] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp. 598–609, Virginia, Va, USA, November 2007.

[3] G. Ateniese, S. Kamara, and J. Katz, "Proofs of storage from homomorphic identification protocols," in *Proceedings of the International Conference on Theory and Application of Cryptology and Information Security: Advances in Cryptology*, vol. 5912, pp. 319–333, 2009.

[4] R. Lu, X. Lin, T. H. Luan, X. Liang, and X. Shen, "Pseudonym changing at social spots: an effective strategy for location privacy in VANETs," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 1, pp. 86–96, 2012.

[5] N. Kaaniche, A. Boudguiga, and M. Laurent, "ID-based cryptography for secure cloud data storage," in *Proceedings of the*

IEEE Sixth International Conference on Cloud Computing, pp. 375–382, 2013.

[6] Q.-A. Wang, C. Wang, K. Ren, W.-J. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 847–859, 2011.

[7] J. Yuan and S. Yu, "Public integrity auditing for dynamic data sharing with multiuser modification," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 8, pp. 1717–1726, 2015.

[8] K. Zeng, "Publicly verifiable remote data integrity," in *Proceedings of the 10th International Conference on Information and Communications Security*, pp. 419–434, 2008.

[9] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 12, pp. 2231–2244, 2012.

[10] Y. Zhu, H. Wang, Z. Hu, G. J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in *Proceedings of the 26th Annual ACM Symposium on Applied Computing (SAC '11)*, pp. 1550–1557, March 2011.

[11] L. Xue, J. Ni, Y. Li, and J. Shen, "Provable data transfer from provable data possession and deletion in cloud storage," *Computer Standard & interfaces*, March 14, 2016.

[12] H. Jin, K. Zhou, H. Jiang, D. Lei, R. Wei, and C. Li, "Full integrity and freshness for cloud data," *Future Generation Computer Systems*, 2016.

[13] H. Wang, J. Domingo-Ferrer, Q. Wu, and B. Qin, "Identity-based remote data possession checking in public clouds," *IET Information Security*, vol. 8, no. 2, pp. 114–121, 2014.

[14] J. Zhang and Q. Dong, "Efficient ID-based public auditing for the outsourced data in cloud storage," *Information Sciences*, vol. 343-344, pp. 1–14, 2016.

[15] Y. Yu, L. Xue, M. H. Au et al., "Cloud data integrity checking with an identity-based auditing mechanism from RSA," *Future Generation Computer Systems*, vol. 62, pp. 85–91, 2016.

[16] L. Wei, H. Zhu, Z. Cao et al., "Security and privacy for storage and computation in cloud computing," *Information Sciences*, vol. 258, pp. 371–386, 2014.

[17] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proceedings of the IEEE INFO-COM*, pp. 525–533, March 2010.

[18] S. Worku, C. Xu, J. Zhao, and X. He, "Secure and efficient privacy-preserving public auditing scheme," *Computer and Electrical Engineering*, vol. 40, pp. 1703–1713, 2014.

[19] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on computers*, vol. 62, no. 2, pp. 362–375, 2013.

[20] J. Zhang and X. Zhao, "Privacy-preserving public auditing scheme for shared data with supporting multi-function," *Journal of Communications*, vol. 10, no. 7, pp. 535–542, 2015.

[21] Y. Li, Y. Yu, B. Yang, G. Min, and H. Wu, "Privacy preserving cloud data auditing with efficient key update," *Future Generation Computer Systems*, 2016.