On the Security of Three Public Auditing Schemes in Cloud Computing

Yang Ming[1] and Yumin Wang[2]
(Corresponding author: Yang Ming)
School of Information Engineering, Chang'an University[1]
Xi'an, Shaanxi 710064, China
(Email: yangming@chd.edu.cn)
State Key Laboratory of ISN, Xidian University[2]
Xi'an, Shaanxi 710071, China
(Received Apr. 13, 2015; revised and accepted May 20 & June 4, 2015)

Abstract

Cloud computing provides a scalability environment for growing amounts of data and processes that work on various applications and services by means of on-demand self-services. It is necessary for cloud service provider to offer an efficient audit service to check the integrity and availability of the stored data in cloud. In this paper, we study three auditing schemes for stored data including the public auditing scheme with user revocation, the proxy provable data possession and the identity-based remote data possession checking. All three mechanisms claimed that their schemes satisfied the security property of correctness. It is regretful that this comment shows that an active adversary can arbitrary alter the cloud data to generate the valid auditing response which can pass the verification. Then, we discussed the origin of the security flaw and proposed methods to remedy the weakness. Our work can help cryptographers and engineers design and implement more secure and efficient auditing mechanism in the cloud.

Keywords: Bilinear pairings, cloud computing, provable data possession, storage auditing

1 Introduction

In recent years, cloud computing is a promising computing model that enables convenient and on-demand network access to a shared pool of computing resources [12]. It provided a flexible, dynamic, resilient and cost effective infrastructure for both academic and business environments, it rapidly expands as an alternative to conventional office-based computing. Cloud computing offers various types of services, including, Infrastructure as a Service (IaaS, Amazon's Elastic Cloud), Platform as a Service (PaaS, Microsoft Azure) and Software as a Service (SaaS, Google Web Mail Service) [1]. The cloud storage is an important service of cloud computing, which allows data owners to move data from their local computing system to the cloud. Thus, it relieves the burden for storage management and maintenance for the data owners.

Although cloud storage service (CSS) provided many appealing benefits for the user, it also prompts a number of security issues, because the user data or archives are stored into an uncertain storage pool outside the enterprises. Firstly, data owners would worry their data could be mis-used or accessed by unauthorized users. Secondly, the data owners would worry their data could be lost in the cloud. Therefore, it is necessary for cloud service providers to offer an efficient audit service to check the integrity and availability of the stored data in the cloud.

The traditional cryptographic technologies for data integrity and availability, based on hash functions and signature schemes, cannot work on the outsourced data without a local copy of data. In addition, it is not a practical solution for data validation by downloading them due to the expensive communications, especially for large size files. Therefore, it is crucial to realize public auditability, so that data owners may resort to a third party auditor (TPA), who has expertise and capabilities that a common user does not have, for periodically auditing the outsourced data.

To achieve this goal, two novel approaches called provable data possession (PDP) [2] and proofs of retrievability (POR) [7]. The new techniques are such a probabilistic proof technique for a storage provider to prove the integrity and ownership of user's data without download data. In 2007, Ateniese et al. [2] firstly proposed the PDP model for ensuring possession of files on untrusted storages and provided an RSA-based scheme for the static case They also proposed a publicly verifiable version, which allows anyone, not just the owner, to challenge the servers for data possession. In 2008, Ateniese et al. [3] proposed a dynamic PDP called scalable PDP which can support dynamic data operations. The new

scheme is constructed using cryptographic Hash function and symmetric key encryption, but the number of updates and challenges is limited and need to be prefixed and block insertion is not allowed. Since then, Erway et al. [6] introduced two dynamic PDP schemes based on a hash function tree. In 2007, Juels [7] presented a POR method to audit the integrity of data. However, it relies largely on preprocessing steps the user conducts before sending the data to cloud service provider (CSP), which prevent any efficient extension to update data. Shacham and Waters [11] proposed the compact POR (CPOR) scheme built from BLS signature [4] with proofs of security. They also use publicly verifiable homomorphic linear authentications that are built form provable secure BLS signature. Wang et al. [22] presented a dynamic scheme with by integrating above CPOR scheme and Merkle Hash Tree in DPDP. Wang et al. [17] provided a privacy-preserving auditing protocol. This scheme achieves batch auditing to support efficient handling of multiple auditing tasks. In 2011, based on fragment structure, random sampling and index-hash table, Zhu et al. [28] propose a dynamic audit service for verifying the integrity of an untrusted and outsourced storage, which supporting provable updates to outsourced data, and timely abnormal detection. Since then, many other auditing mechanisms such as [13, 14, 15, 16, 18, 19, 23, 24, 25, 26, 27] have been proposed for protecting the integrity of the data in the cloud.

In 2013, Wang boyang et al. [15] proposed a novel public auditing mechanism for the integrity of shared data with efficient user revocation. By utilizing proxy re-signatures, it allows the cloud to re-sign blocks on behalf of existing users during user revocation phase, so that existing users do not need to download and re-sign blocks by themselves. In addition, a public verifier is always able to audit the integrity of shared data without retrieving the entire data from the cloud, even if some part of shared data has been re-signed by the cloud. Wang [19] proposed the concept of proxy provable data possession (PPDP), which is a matter of crucial importance when the user can not perform the remote data possession checking. Based on bilinear pairings, he gives an efficient and provable secure PPDP protocol. In 2014, Wang et al. [20] firstly formalized the model of identity based remote data possession checking (ID-RDPC) protocol for secure cloud storage. Based on bilinear pairings, they proposed the first concrete ID-RDPC protocol which was proven secure under the CDH assumption.

In this paper, we study the above three auditing mechanisms for secure cloud storage, including public auditing mechanism with user revocation [15], proxy provable data possession [19] and identity based remote data possession checking [20]. We show that the three schemes do not satisfy the property of correctness. When the active adversaries can arbitrarily tamper the cloud data and produce a valid auditing response to pass the verification. Therefore, the adversaries can cheat the auditor to believe that the data in cloud are well-maintained while in fact the data have been modified. Then, we discuss the origin of the security flaws and give a solution.

2 Preliminaries

In this section, we briefly review the basic concepts on bilinear pairings and the related system models.

2.1 Bilinear Pairings

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative cyclic groups of prime order p and let g be a generator of $\mathbb{G}_1$. The map e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is said to be an admissible bilinear pairing with the following properties:

1) **Bilinearity:** $e(u^a, v^b) = e(u, v)^{ab}$ for all $u, v \in \mathbb{G}_1$ and for all $a, b \in \mathbb{Z}_p$;

2) **Non-degeneracy:** $e(g, g) \neq 1_{\mathbb{G}_1}$;

3) **Computability:** There exists an efficient algorithm to compute $e(u, v)$ for all $u, v \in \mathbb{G}_1$.

We note the modified Weil and Tate pairings associated with supersingular elliptic curves are examples of such admissible pairings.

2.2 System Model

The system model of public auditing mechanism with user revocation can be shown in Figure 1 [15]. The three entities: the cloud, the third party auditor (TPA), and users are involved in public auditing mechanism. The cloud offers data storage and sharing services to users. The TPA is able to publicly audit the integrity of shared data in the cloud for users. In a group, there is one original user and a number of group users. The original user is the original owner of data. This original user creates and shares data with other users in the group through the cloud. Both the original user and group users are able to access, download and modify shared data. Shared data is further divided into a number of blocks. A user can modify a block in shared data by performing an insert, delete or update operation on the block.

The system model of proxy provable data possession (PPDP) can be shown in Figure 2 [19]. The PPDP system consists of three different entities: user, public cloud service (PCS), and proxy. The user moves the massive data to the remote PCS, the PCS has significant storage space and computation resource to maintain the users' data; the proxy, which is delegated to check user's data possession.

The system model of identity based remote data possession checking (ID-RDPC) can be shown in Figure 3 [20]. The ID-RDPC protocol consists of three different entities: private key generator (PKG), public cloud service (PCS) and client. The PKG generates the public parameters and master public key and client's private key. The client moves the data to be stored on the public cloud

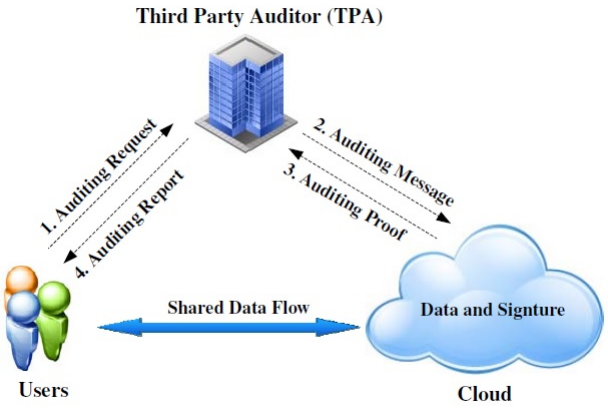

Figure 1: System model of public auditing mechanism

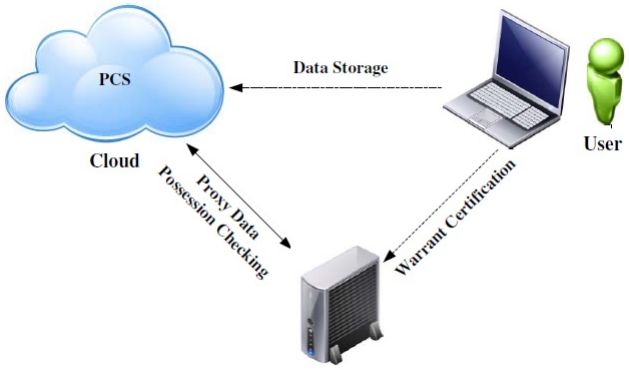

Figure 2: System model of PPDP

for maintenance and computation. The PCS has significant storage space and computation resource to maintain the users' data.

3 The Security of Three Auditing Mechanisms

In this section, we firstly review public auditing scheme with user revocation [15], proxy provable data possession [19] and identity based remote data possession checking [20], and then give the security analysis of three mechanisms in active adversaries.

3.1 Overview of Wang et al.'s Scheme

This scheme consists of six procedures: **KeyGen**, **ReKey**, **Sign**, **ReSign**, **ProofGen** and **ProofVerify**. The reader can refer to [15] for detailed description.

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of order p, g be a generator of $\mathbb{G}_1$, e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a bilinear map, ω be a random element of $\mathbb{G}_1$. The global parameters are $\{e, p, \mathbb{G}_1, \mathbb{G}_2, g, \omega, H, H'\}$, where H is a hash function with H: $\{0,1\}^* \rightarrow \mathbb{G}_1$ and H' is a hash function with H': $\{0,1\}^* \rightarrow \mathbb{Z}_p$. The total number of blocks in shared data is n, and shared data is described as $M = (m_1, \cdots, m_n)$.

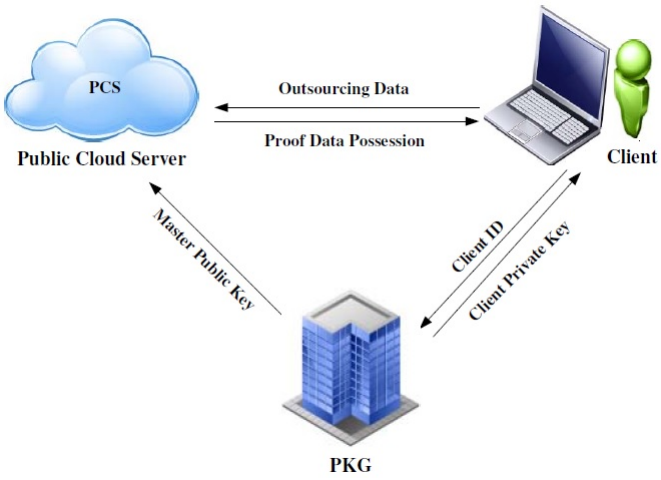

Figure 3: System model of ID-RDPC

The total number of users in the group is d.

KeyGen. For user u_i, he/she generates a random $x_i \in \mathbb{Z}_p$, and outputs his/her public key $pk_i = g^{x_i}$ and private key $sk_i = x_i$. Without loss of generality, we assume user u_1 is the original user, who is the creator of shared data. The original user also creates a user list (UL), which contains ids of all the users in the group. The user list is public and signed by the original user.

ReKey. The cloud generates a re-signing key $rk_{i\rightarrow j}$ as follows:

1) The cloud generates a random $r \in \mathbb{Z}_p$ and sends it to user u_i;
2) User u_i sends r/x_i to user u_j, where $sk_i = x_i$;
3) User u_j sends rx_j/x_i to the cloud, where $sk_j = x_j$;
4) The cloud recovers $rk_{i\rightarrow j} = x_j/x_i$.

Sign. Given private key $sk_i = x_i$, block $m_k \in \mathbb{Z}_p$ in shared data M and its block identifier id_k, where $k \in [1, n]$, the user u_i outputs the signature on block m_k as $\sigma_k = (H(id_k)\omega^{m_k})^{x_i}$.

ReSign. When user u_i is revoked from the group, the cloud is able to convert signatures of user u_i into signatures of user u_j on the same block. More specifically, given re-signing key $rk_{i\rightarrow j}$, public key pk_i, signature σ_k, block m_k and block identifier id_k, the cloud first checks that $e(\sigma_k, g) = e(H(id_k)\omega^{m_k}, pk_i)$. If the verification result is 0, the cloud outputs $\perp$; otherwise, it outputs $\sigma'_k = \sigma_k^{rk_{i\rightarrow j}} = (H(id_k)\omega^{m_k})^{x_i \cdot x_j/x_i} = (H(id_k)\omega^{m_k})^{x_j}$. After the re-signing, the original user removes user u_i's id from UL and signs the new UL.

ProofGen. To audit the integrity of shared data, the TPA generates an auditing message as follows:

1) Randomly picks a c-element subset L of set $[1, n]$ to locate the c selected random blocks that will be checked in this auditing task;
2) Generates a random $y_l \in \mathbb{Z}_p$, for $l \in L$ and q is a much smaller prime than p;
3) Outputs an auditing message $\{(l, y_l)\}_{l\in L}$, and sends it to the cloud.

After receiving an auditing message, the cloud generates a proof of possession of shared data M. More concretely,

1) The cloud divides set L into d subset $L_1, L_2, \cdots, L_d$, where L_i is the subset of selected blocks signed by user u_i. And the number of elements in subset L_i is c_i. Clearly, we have $c = \sum_{i=1}^{d} c_i$, $L = L_1 \cup \cdots \cup L_d$ and $L_i \cap L_j = \Phi$ for $i \neq j$;
2) For each set L_i, the cloud computes $\alpha_i = \sum_{l\in L_i} y_l m_l \in \mathbb{Z}_p$ and $\beta_i = \prod_{l\in L_i} \sigma_l^{y_l} \in \mathbb{G}_1$;
3) The cloud outputs an auditing proof $\{\alpha, \beta, \{id_l\}_{l\in L}\}$, and sends it to the verifier, where $\alpha = (\alpha_1, \cdots, \alpha_d)$ and $\beta = (\beta_1, \cdots, \beta_d)$.

ProofVerify. With an auditing proof $\{\alpha, \beta, \{id_l\}_{l\in L}\}$, an auditing message $\{(l, y_l)\}_{l\in L}$, and all the existing users'public keys $(pk_1, \cdots, pk_d)$, the TPA checks the correctness of this auditing proof as

$$e(\prod_{i=1}^{d} \beta_i, g) = \prod_{i=1}^{d} e(\prod_{l\in L_i} H(id_l)^{y_l} \cdot \omega^{\alpha_i}, pk_i).$$

If the result is 1, the verifier believes that the integrity of all the blocks in shared data M is correct. Otherwise, the verifier outputs 0.

3.2 Security Analysis on Wang et al.'s Scheme

As the audit scheme for shared data with efficient user revocation in cloud, Wang boyang et al.'s scheme enjoys many desirable security properties. Informally, this mechanism needs that it be infeasible to fool the TPA into accepting false statements, i.e. the TPA is able to correctly detect whether there is any corrupted data.

However, we show that when an active adversary, such as a bug planted in the software running on the cloud server by a malicious programmer or a hacker, is involved in the auditing process. Specifically, the adversary can arbitrarily modify or tamper the outsourced data and fool the TPA to believe the data are well preserved in the cloud.

All the information the adversary has to know is how the data are modified. The details are described as follows:

1) For each set L_i, the adversary $\mathcal{A}$ firstly modifies the block m_l to $m_l^* = m_l + f_l$ for $l \in L_i$ and records the values f_l, i.e. how the shared data are modified.
2) When getting the challenge $\{(l, y_l)\}_{l\in L}$ from the TPA, the cloud honestly executes **ProofGen** algorithm to compute $\alpha^* = (\alpha_1^*, \cdots, \alpha_d^*), \beta = (\beta_1, \cdots, \beta_d)$ as follows: For $l \in L_i$, the cloud computes

$$\begin{aligned}\alpha_i^* &= \sum\nolimits_{l\in L_i} y_l m_l^* = \sum\nolimits_{l\in L_i} y_l(m_l + f_l) \\ &= \alpha_i + \sum\nolimits_{l\in L_i} y_l f_l \\ \beta_i &= \prod\nolimits_{l\in L_i} \sigma_l^{y_l}.\end{aligned}$$

Finally, the cloud sends the auditing proof $(\alpha^*, \beta, \{id_l\}_{l\in L})$ to the TPA.

3) The adversary $\mathcal{A}$ intercepts the auditing proof $(\alpha^*, \beta, \{id_l\}_{l\in L})$ from the cloud to TPA, and modifies each α_i^* to $\alpha_i = \alpha_i^* - \sum_{l\in L_i} y_l f_l$ for $l \in L_i$.

By performing such a modification, the adversary $\mathcal{A}$ derives a correct proof with respect to the original data blocks M, and sends it to the TPA. As a result, in **ProofVerify** phase, the modified auditing proof can make the equation hold and, thus the TPA believes that shared data are all well-maintained, while the data have been polluted by the adversary $\mathcal{A}$.

3.3 Overview of Wang's Scheme

This scheme consists of six procedures: **SetUp**, **TagGen**, **SignVerify**, **CheckTag**, **GenProof** and **CheckProof**. The reader can refer to [19] for detailed description.

Suppose the maximum number of the stored block-tag pairs is n. Let f: $\mathbb{Z}_q^* \times \{1, 2, \cdots, n\} \to \mathbb{Z}_q^*$ and Ω: $\mathbb{Z}_q^* \times \{1, 2, \cdots, n\} \to \mathbb{Z}_q^*$ be two pseudo-random functions, and let π: $\mathbb{Z}_q^* \times \{1, 2, \cdots, n\} \to \{1, 2, \cdots, n\}$ be a pseudo-random permutation and H: $\mathbb{G}_2 \times \{0, 1\} \to \mathbb{Z}_q^*$, h: $\mathbb{Z}_q^* \to \mathbb{G}_1$ be cryptographic hash functions.

Let g be a generator of $\mathbb{G}_1$. We assume that the file F (maybe encoded by using error-correcting code, such as, Reed-Solomon code) is divided into n blocks $(m_1, m_2, \cdots, m_n)$ where $m_i \in \mathbb{Z}_q^*$ and q is the order of $\mathbb{G}_1$ and $\mathbb{G}_2$. Without loss of generality, we denote $F = (m_1, m_2, \cdots, m_n)$.

SetUp. The user picks a random number $x \in \mathbb{Z}_q^*$ as its private key and computes $X = g^x$ as its public key. The PCS picks a random number $y \in \mathbb{Z}_q^*$ as its private key and computes $Y = g^y$ as its public key. The proxy picks a random number $z \in \mathbb{Z}_q^*$ as its private key and computes $Z = g^z$ as its public key. The user picks a random element $u \in \mathbb{G}_1$ and a secure signature/verification algorithm pair $(SigGen, SignVerify)$. Finally, the system parameter is $param = \{\mathbb{G}_1, \mathbb{G}_2, e, f, \Omega, \pi, H, h, X, Y, Z, u, q, (SigGen, SignVerify)\}$.

TagGen. Given $F = (m_1, m_2, \cdots, m_n)$ and the warrant ω, the user generates the tag T_{m_i} of the block m_i as follows:

1) Client computes $t = H(e(Y,Z)^x, \omega)$, $W_i = \Omega_t(i)$;
2) Client computes $T_{m_i} = (h(W_i)u^{m_i})^x$ for $i \in [1,n]$;
3) Client computes the signature $Sign = SigGen_x(\omega)$ on the warrant ω.

Then the user sends the the block-Tag pairs collection $\{(T_{m_i}, m_i), i \in [1,n]\}$ and the warrant ω to the PCS. The PCS stores the block-tag pairs and the warrant ω. The user deletes the block-tag pairs $\{(T_{m_i}, m_i), i \in [1,n]\}$ from its local storage. At the same time, the user sends the warrant-signature pair $(\omega, Sign)$ to the proxy.

SignVerify. Upon receiving (ω, $Sign$), the proxy performs the verification algorithm $SignVerify(\omega, Sign, X)$. If it is valid, the proxy accepts this warrant ω; otherwise, the proxy rejects it and queries the user for new warrant-certification pair.

CheckTag. Given $\{(T_{m_i}, m_i), i \in [1,n]\}$, PCS performs the procedures for every $i, 1 \le i \le n$, as follows:

1) PCS computes $\hat{t} = H(e(X,Z)^y, \omega)$ and $\hat{W}_i = \Omega_{\hat{t}}(i)$;
2) PCS verifies whether $e(T_i, g) = e(h(\hat{W}_i)u^{m_i}, X)$ holds.

If it holds, then PCS accepts it. Otherwise, PCS rejects it and queries the user for new block-tag pair.

GenProof. Let the challenge be $chal = (c, k_1, k_2)$ where $1 \le c \le n$, $k_1 \in \mathbb{Z}_q^*$, $k_2 \in \mathbb{Z}_q^*$. In this phase, the proxy asks the PCS for remote data possession proof of c file blocks whose indexes are randomly chosen using a pseudo-random permutation keyed with a fresh randomly chosen key for each challenge. On the other hand, the proxy sends $(\omega, Sign)$ to PCS. PCS verifies whether the signature $Sign$ is valid. If it is valid, PCS compares this ω with its stored warrant ω'. When $\omega = \omega'$ and the proxy's queries comply with the warrant ω, PCS performs the procedures as follows. Otherwise, PCS rejects the proxy's query.

1) For $1 \le j \le c$, PCS computes the indexes and coefficients of the blocks for which the proof is generated: $i_j = \pi_{k_1}(j)$, $a_j = f_{k_2}(j)$;
2) PCS computes $T = \prod_{j=1}^{c} T_{m_i}^{a_j}$, $\hat{m} = \sum_{j=1}^{c} a_j m_{i_j}$.

PCS outputs $V = (\hat{m}, T)$ and sends V to the proxy as the response.

CheckProof. Upon receiving the response V from PCS, the proxy performs the procedures as follows:

1) Proxy computes $t = H(e(X,Y)^z, \omega)$;
2) Proxy checks whether the following formula holds

$$e(T,g) = e(\prod_{i=1}^{c} h(\Omega_t(\pi_{k_1}(i)))^{f_{k_2}(i)} u^{\hat{m}}, X).$$

If it holds, then the proxy outputs "success". Otherwise the proxy outputs "failure".

3.4 Security Analysis on Wang's Scheme

As the proxy provable data possession (PPDP) in public cloud, Wang's scheme enjoys many desirable security properties of a PPDP scheme including correctness and unforgeability. Informally, correctness property means that it be infeasible to fool the proxy into accepting false statements, i.e. the proxy is able to correctly detect whether there is any corrupted data.

Similar to the analysis of Wang boyang et al.'s scheme, we show that the Wang's scheme is not secure in active adversary, who can arbitrarily modify or tamper the outsourced data and fool the proxy to believe the data are well preserved without being detected by the proxy in the cloud. The details are described as follows:

1) The adversary $\mathcal{A}$ firstly modifies the block $m_{i,j}$ to $m_{i,j}^* = m_{i,j} + d_{i,j}$ for $i \in [1,n]$, $j \in [1,c]$ and records the values $d_{i,j}$.
2) When PCS receiving the challenge $chal = (c, k_1, k_2)$ from the proxy, the PCS honestly executes **GenProof** algorithm to compute $V^* = (\hat{m}^*, T)$ as follows:

$$\begin{aligned} \hat{m}^* &= \sum_{j=1}^{c} a_j m_{i_j}^* = \sum_{j=1}^{c} a_j (m_{i,j} + d_{i,j}) \\ &= \hat{m} + \sum_{j=1}^{c} a_j d_{i,j} \\ T &= \prod_{j=1}^{c} T_{m_{i_j}}^{a_j}. \end{aligned}$$

Then it sends the $V^* = (\hat{m}^*, T)$ auditing proof to the proxy.

The adversary $\mathcal{A}$ intercepts the auditing proof $V^* = (\hat{m}^*, T)$ and modifies $\hat{m}^*$ to $\hat{m} = \hat{m}^* - \sum_{j=1}^{c} a_j d_{i,j}$. By performing such a modification, the adversary $\mathcal{A}$ obtains a correct proof with respect to the original data $m_{i,j}$ for $i \in [1,n]$ and $j \in [1,c]$. As a result, the proof can pass the auditing verification, which makes the proxy believe that the shared data are well maintained by the PCS, while in fact the data have been corrupted.

3.5 Overview of Wang et al.'s Scheme

An ID-RDPC protocol is a collection of five polynomial-time algorithms: **Setup**, **Extract**, **TagGen**, **GenProof** and **CheckProof**. The reader can refer to [20] for detailed description.

Setup. PKG chooses a random number $x \in \mathbb{Z}_q^*$ and sets $Y = g^x$, where g is a generator of the group $\mathbb{G}_1$. PKG chooses a random element $u \in \mathbb{G}_1$. Define two cryptographic hash functions: H: $\{0,1\} \to \mathbb{Z}_q^*$, h: $\mathbb{Z}_q^* \to \mathbb{G}_1$. Let f be a pseudo-random function and let π be a pseudo-random permutation f: $\mathbb{Z}_q^* \times \{1,2,\cdots,n\} \to$

$\mathbb{Z}_q^*$, π: $\mathbb{Z}_q^* \times \{1, 2, \cdots, n\} \to \{1, 2, \cdots, n\}$. PKG publishes $(\mathbb{G}_1, \mathbb{G}_2, e, q, g, Y, u, H, h, f, \pi)$ and keeps x as the master key.

Extract. A client submits the identity ID to the PKG. The PKG picks $r \in \mathbb{Z}_q^*$ and computes $R = g^r, \sigma = r + xH(ID, R) \bmod q$. The PKG sends the private key $sk_{ID} = (R, \sigma)$ to the client by a secure channel. The client can verify the correctness of the received private key by checking whether $g^\sigma = R \cdot Y^{H(ID,R)}$ holds. If the previous equality holds, the client accepts the private key; otherwise, he/she rejects it.

TagGen. We assume that the client generates the tags sequentially according to the counter i. That is, the client generates a tag for a message block m_2 after m_1, which implies that the client maintains the latest value of the counter i. For m_i, the client performs the TagGen procedure as follows:

1) Compute $T_i = ((h(i)u^{m_i})^\sigma$;
2) Output T_i and send to the PCS.

GenProof. In this phase, the verifier (who can be the client himself/herself) queries the PCS for a proof of data possession of c file blocks whose indices are randomly chosen using a pseudo-random permutation keyed with a fresh random-chosen key for each challenge.

The number $k_1 \in Z_q^*$ is the random key of the pseudo-random permutation π. Also, $k_2 \in Z_q^*$ is the random key of the pseudo-random function f. Let the challenge be $chal = (c, k_1, k_2)$. Then, the PCS does:

1) For $1 \leqslant j \leqslant c$, compute the indices and coefficients of the blocks for which the proof is generated: $i_j = \pi_{k_1}(j)$, $a_j = f_{k_2}(j)$. In this step, the challenge $chal$ defines an ordered set $\{c, i_1, \cdots, i_c, a_1, \cdots, a_c\}$;
2) Compute $T = \prod_{j=1}^{c} T_{i_j}^{a_j}$, $\hat{m} = \sum_{j=1}^{c} a_j m_{i_j}$;
3) Output $V = (T, \hat{m})$ and send V to the client as the response to the $chal$ query.

CheckProof. Upon receiving the response V from the PCS, the verifier (who can be the client himself/herself) does:

1) Check the equation: $e(T, g) = e\left(\prod_{i=1}^{c} h\left(\pi_{K_1}(i)\right)^{f_{k_2}(i)} u^{\hat{m}}, R \cdot Y^{H(ID,R)}\right)$.
2) If the previous equation holds, output "success". Otherwise, output "failure".

3.6 Security Analysis on Wang et al.'s Scheme

As an Identity based remote data possession checking (ID-RDPC) protocol in the public clouds, Wang et al.'s scheme [20] enjoys many desirable security properties. Informally, this mechanism needs that it be infeasible to fool the client into accepting false statements, i.e. the client is able to correctly detect whether there is any corrupted data.

However, we show that when an active adversary, such as a bug planted in the software running on the cloud server by a malicious programmer or a hacker, is involved in the remote data possession checking process. Specifically, the active adversary can arbitrarily modify or tamper the outsourced data and fool the PCS to believe the data are well preserved in the cloud.

All the information the adversary has to know is how the data are modified. The details are described as follows:

1) The adversary $\mathcal{A}$ firstly modifies the block m_{i_j} to $m_{i_j}^* = m_{i_j} + d_{i_j}$ for $j \in [1, c]$ and records the values d_{i_j}.
2) When the PCS receiving the challenge $chal = (c, k_1, k_2)$ from the client, the PCS honestly executes **GenProof** algorithm to compute $V^* = (\hat{m}^*, T)$ as follows:

$$\begin{aligned} \hat{m}^* &= \sum\nolimits_{j=1}^{c} a_j m_{i_j}^* = \sum\nolimits_{j=1}^{c} a_j (m_{i,j} + d_{i,j}) \\ &= \hat{m} + \sum\nolimits_{j=1}^{c} a_j d_{i,j} \\ T &= \prod\nolimits_{j=1}^{c} T_{m_{i_j}}^{a_j}. \end{aligned}$$

Then it sends the response $V^* = (\hat{m}^*, T)$ to the client.

The adversary $\mathcal{A}$ intercepts the response $V^* = (\hat{m}^*, T)$ and modifies $\hat{m}^*$ to $\hat{m} = \hat{m}^* - \sum_{j=1}^{c} a_j d_{i_j}$. By performing such a modification, the adversary $\mathcal{A}$ obtains a correct proof with respect to the original data m_{i_j} for $j \in [1, c]$. As a result, the proof can pass the verification, which makes the client believe that the shared data are well maintained by the PCS, while in fact the data have been corrupted.

3.7 Discussion on the Origin and Method

Through the above analysis, it is known Wang boyang et al.'s scheme, Wang's scheme and Wang et al.'s scheme can not satisfy the correctness. Taking use of our proposed attack methods, an active adversary can arbitrarily modify or tamper the outsourced data in the cloud and fool the auditor (TPA, Proxy and client) to believe the data are well preserved in the cloud.

Why does the above three schemes have security flaw? There is no an authentication check on the auditing response about the challenge. This property incurs the security flaws. It is important to clarity the security in order to design secure and practical auditing mechanism in cloud storage.

To solve the security problem, we can use the technique of digital signature. Specifically, in auditing response, the cloud service firstly uses the private key to compute a

signature for proof and, then send both the proof and the corresponding signature as the response to the challenge. Receiving the response, the auditor first verifies whether the signature is valid or not. If it is valid, the auditor performs the auditing verification protocol; otherwise, the auditor discards the response. Therefore, if the shared data have been modified, the auditor must be able to detect it because of the employment of the digital signatures.

We can use HMAC technique [8], which also provided message authentication function. Meantime, the 3-move protocol [25, 26, 27]: commitment, challenge and response (for example Schnorr's protocol [10]) is also used in auditing response and prevent to the pollution attacks from active adversaries.

4 Conclusion

With the development of cloud computing, many people focused on the study of information security in Cloud Environments [5, 9, 21]. Wang boyang et al. [15], Wang [19] and Wang et al. [20] proposed a secure auditing mechanisms for the stored data in cloud. However, through cryptanalysis, we show that their schemes still has security weakness. By giving a concrete attacks, we prove that the two schemes are not secure in active adversary environment. Specially, any adversary can modify the data without being detected by the auditor in the verification phase. Finally, we study the origin of the security flaw and propose a solution to remedy this weakness.

Acknowledgments

This work was supposed by the National Natural Science Foundation of China (No.61202438), the China Postdoctoral Science Foundation (No.2011M501427), the Key Project of Industry Science and Technology of Shaanxi Province (No.2015GY014) and the Project of Technology Transfer Promoting Engineering of Xi'an City (No.CXY1437(10)).

References

[1] M. Armbrust, et al, "A view of cloud computing," *ACM Communication*, vol. 53, pp. 50–58, 2010.

[2] G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, and D. X. Song, "Provable data possession at untrusted stores," in *Proceedings of the 2007 ACM Conference on Computer and Communications Security (CCS'07)*, pp. 598–609, 2007.

[3] G. Ateniese, R. D. Pietro, L. V. Mancini, G. Tsudik, "Scalable and efficient provable data possession," in *Proceedings of the 4th ACM International Conference on Security and Privacy in Communication Networks (SecureComm'08)*, pp. 1–10, 2008.

[4] D. Boneh, B. Lynn, H. Shacham, "Short signatures from the Weil pairing," *Journal of Cryptology*, vol. 17, no. 4, pp. 297–319, 2004.

[5] P. S. Chung, C. W. Liu and M. S. Hwang, "A study of attribute-based proxy re-encryption scheme in cloud environments," *International Journal of Network Security*,vol. 16, no. 1, pp. 1–13, 2014.

[6] C. C. Erway, A. Kpc, C. Papamanthou, R. Tamassia, "Dynamic provable data possession," in *Proceedings of the 2009 ACM Conference on Computer and Communications Security (CCS'09)*, pp. 213–222, 2009.

[7] A. Juels and B. S. K. Jr, "PORS: Proofs of retrievability for large files," in *Proceedings of the 2007 ACM Conference on Computer and Communications Security (CCS'07)*, pp. 584–597, 2007.

[8] H. Krawczyk, M. Bellare and R. Canetti, *HMAC: Keyed-hashing for Message Authentication*, RFC 2104, Feb. 1997.

[9] C. C. Lee, P. S. Chung and M. S. Hwang, "A survey on attribute-based encryption schemes of access control in cloud environments," *International Journal of Network Security*, vol. 15, no. 4, pp. 231–240, 2013.

[10] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptology (Crypto'89)*, LNCS 435, pp. 239–252, 1990.

[11] H. Shacham, B. Waters, "Compact proofs of retrievability," in *Advances in Cryptology (Asiacrypt'08)*, LNCS 5350, pp. 90–107, 2008.

[12] L. M. Vaquero, L. Rodero-Merino, J. Caceres, M. Lindner, "A break in the clouds: Towards a cloud definition," *ACM SIGCOMM Computer Communication Review*, vol. 39, pp. 50–55, 2009.

[13] B. Wang, B. Li, H. Li, "ORUTA: Privacy-preserving public auditing for shared data in the cloud," in *Proceedings of the IEEE International Conference on Cloud Computing*, pp. 293–302, 2012.

[14] B. Wang, B. Li, H. Li, "KNOX: Privacy-preserving auditing for shared data with large groups in the cloud," in *Applied Cryptography and Network Security (ACNS'12)*, LNCS 7341, pp. 507–525, 2012.

[15] B. Wang, B. Li and H. Li, "Public auditing for shared data with efficient user revocation in the cloud," in *Proceedings of IEEE INFOCOM'13*, pp. 2904–2912, 2013.

[16] C. Wang, K. Ren, W. Lou, J. Li, "Toward publicly auditable secure cloud data storage services," *IEEE Networks*, vol. 24, no. 4, pp. 19–24, 2010.

[17] C. Wang, Q. Wang, K. Ren, W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proceedings IEEE of INFOCOM'10*, pp. 1–9, 2010.

[18] C. Wang, Q. Wang, K. Ren, N. Cao, W. Lou, "Toward secure and dependable storage services in cloud computing," *IEEE Transactions on Services Computing*, vol. 5, no. 2, pp. 220–232, 2012.

[19] H. Wang, "Proxy provable data possession in public clouds," *IEEE Transactions on Services Computing*, vol. 6, no. 4, pp. 551–559, 2013.

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

[21] J. Wang, X. Yu and M. Zhao, "Fault-tolerant verifiable keyword symmetric searchable encryption in hybrid cloud," *International Journal of Network Security*, vol. 17, no. 4, pp. 471–483, 2015.

[22] Q. Wang, C. Wang, J. Li, K. Ren, W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *Proceedings of the 14th European Symposium on Research in Computer Security (ESORICS'09)*, LNCS 5789, pp. 355–370, 2009.

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

[24] K. Yang, X. Jia, "Data storage auditing service in cloud computing: Challenges, methods and opportunities," *World Wide Web*, vol. 15, no. 4, pp. 409–428, 2012.

[25] Y. Zhu, G. J. Ahn, H. Hu, S. S. Yau, Ho G. An and C. J. Hu, "Dynamic audit services for outsourced storages in Clouds," *IEEE Transactions on Services Computing*, vol. 6, no. 2, pp. 227–238, 2013.

[26] Y. Zhu, H. Hu, G. J. Ahn, S. S. Yau, "Efficient audit service outsourcing for data integrity in clouds", *The Journal of Systems and Software*, vol. 85, pp. 1083–1095, 2012.

[27] Y. Zhu, H. Hu, G. J. Ahn, M. Yu, "Cooperative provable data possession for integrity verification in multi-cloud storage," *IEEE Transactions on Parallel Distribute System*, vol. 23, no. 12, pp. 2231–2244, 2012.

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

Yang Ming was born in Shaanxi Province, China in 1979. He received the B.S. and M.S. degrees in mathematics from Xian University of Technology in 2002 and 2005 respectively, and the Ph.D. degree in cryptography from Xidian University in 2008. Currently he is a supervisor of postgraduate and associate professor of Chang'an University. His research interests include cryptography and digital signature.

Yumin Wang was born in Beijing, China in 1936. He received the B.S. degree from the Department of Telecommunication Engineering, Xidian University in 1959. In 1979-1981, he was a visiting scholar in Department of Electronic Engineering, Hawaii University. Currently he is a doctoral supervisor and professor of Xidian University. He is a fellow member of the Board of Governors of the Chinese Institute of Cryptography (preparatory committee) and a Senior Member (SM) of IEEE. His research interests include information theory, coding, and cryptography.