An Efficient Provable Secure Public Auditing Scheme for Cloud Storage

Chunxiang Xu, Yuan Zhang, Yong Yu, Xiaojun Zhang and Junwei Wen
School of Computer Science and Engineering, University of Electronic Science and Technology of China, 2006 Xi Yuan Avenue, West High-tech Zone, Chengdu 611731, China

Received July 23, 2014; accepted September 10, 2014; published November 30, 2014

Abstract

Cloud storage provides an easy, cost-effective and reliable way of data management for users without the burden of local data storage and maintenance. Whereas, this new paradigm poses many challenges on integrity and privacy of users' data, since users losing grip on their data after outsourcing the data to the cloud server. In order to address these problems, recently, Worku et al. have proposed an efficient privacy-preserving public auditing scheme for cloud storage. However, in this paper, we point out the security flaw existing in the scheme. An adversary, who is on-line and active, is capable of modifying the outsourced data arbitrarily and avoiding the detection by exploiting the security flaw. To fix this security flaw, we further propose a secure and efficient privacy-preserving public auditing scheme, which makes up the security flaw of Worku et al.'s scheme while retaining all the features. Finally, we give a formal security proof and the performance analysis, they show the proposed scheme has much more advantages over the Worku et al.'s scheme.

Keywords: cloud storage, auditing scheme, cryptanalysis, improvement

http://dx.doi.org/10.3837/tiis.2014.11.032

1. Introduction

Cloud storage is a momentous service of cloud computing, which provides an easy, cost-effective and reliable way of data management for users. Using cloud storage service, users can access their data remotely through the internet without incurring substantial hardware, software, and personnel costs involved in deploying and maintaining application in local storage. However, due to users losing grip on their data after outsourcing the data into cloud server, the integrity and correctness of the data are being put at risk and have naturally become the concerned focus of the cloud users. As a semi-trust part for cloud users, cloud server may discard the data motivated by the interest but claim that the data are still correctly stored. Furthermore, an adversary with profits motivation, who is interested in distorting the cloud user's data but convinces the cloud user of the data correctness and integrity [1], [2], [3]. Therefore, it is vital to check the correctness and integrity of the cloud data for protecting the stored data both from external adversaries and the cloud server itself.

Several outstanding research achievements [1], [4], [2], [5] in addressing integrity and correctness of outsourced data have emerged. However, some of them have been proved insecure [6], [7], and some of them still have the room for performance improvement.

Recently, a public auditing scheme [3] was proposed to check the correctness and integrity of outsourced data for cloud storage. This scheme fixes the security flaw pointed out by [6], meantime, it also improves efficiency.

In this paper, we review the public auditing scheme in [3] and point out that the scheme owns a security flaw. With the security flaw, an adversary is able to arbitrarily modify the cloud user's data, and it cannot be discovered. Particularly, an adversary, who is on-line and active, can produce a valid auditing proof to pass the data correctness and integrity checking. Once successful, the adversary can cheat the third-part auditor (TPA) and the cloud user. The adversary just needs recording how data are modified, intercepting, tampering with and transponding an interaction message between the cloud server and TPA to avoid the data correctness and integrity detection. Moreover, we propose a new secure and efficient privacy-preserving public auditing scheme. The proposed scheme fixes the security flaw existing in the Worku et al.'s scheme, while retaining all the features. At last, we give a formal security proof of the proposed scheme, it shows that the scheme is secure and fixes the security flaw indeed. We also give a performance analysis of the proposed scheme to prove the scheme is efficient.

2. Preliminaries

2.1 The System and Security Model

Our system model and security model are designed based on the model in [3].

An auditing system for cloud storage involves cloud user, cloud server and third-party auditor (TPA) as shown in **Fig. 1**.

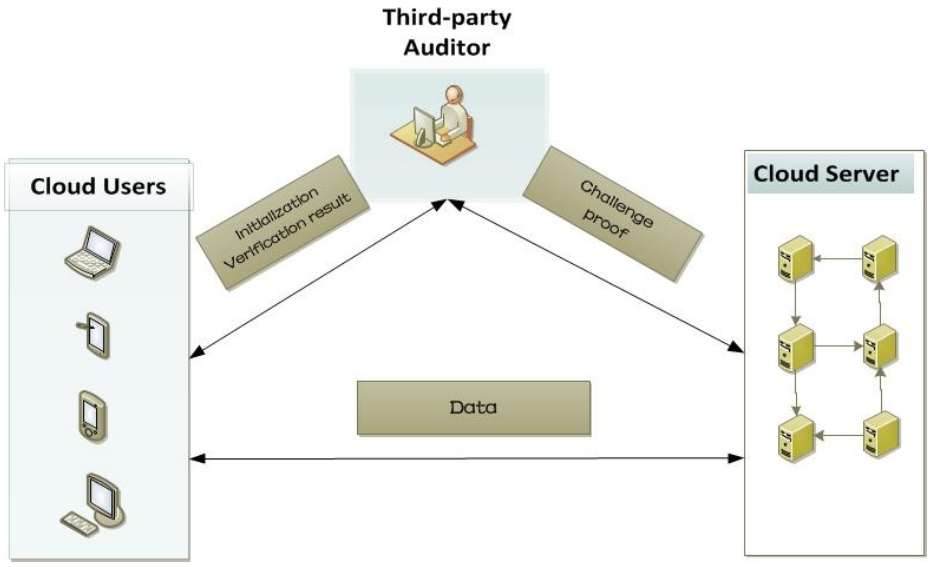


Fig. 1. The System Model

The cloud user is the data owner, who needs flexibly to store and get his data in the cloud.

The cloud server is the provider of cloud services, it has significant storage space and a massive amount of computing power. The cloud server as a semi-trust part for the cloud user, that is, most of the time, cloud server executes the auditing protocol honestly, but in relationship with individual cloud users which are its stakeholders, cloud server might deviate from the prescribed routine.

The TPA has expertise and capabilities that cloud user does not have, who is managed by a trusted organization and will audit the data stored in cloud server by cloud users when needed. The TPA is regarded as an honest entity but curious. That is, the TPA honestly performs the auditing protocol, it is reliable and independent and thus has no incentive to collude with either the cloud server or the users during the auditing process. However, it is interest in the users' data.

An auditing scheme can be said to be secure if and only if both of the following conditions hold:

1. There is no any polynomial time algorithm that can pass the auditing with non-negligible probability.
2. There is a polynomial time extractor that might recover the original data by doing multiple challenge-response executions.

The security of our scheme is constructed under the hardness assumption of computational Diffie-Hellman problem (CDH) and Discrete Logarithm problem (DL) over bilinear groups in the random oracle model [3], [10].

2.2 Notations and Basic Theory

We now introduce some necessary notations and basic theory, which will be utilized below.

We will work in the group Z_p . F denotes the data file and m denotes the data block. $F = \{m_1, m_2, ..., m_n\}$ is made up of data blocks to be stored in cloud server, for each data file, n may be different.

Bilinear Map. Let G and G_T be two multiplicative cyclic groups of the same prime order p, g be a generator of G. A bilinear map is that $e: G \times G \to G_T$ with the following properties:

(1) Linearity. For any $u, u_1, u_2, v, v_1, v_2 \in G$, then

$$e(u_1 \cdot u_2, v) = e(u_1, v) \cdot e(u_2, v)$$

$$e(u, v_1 \cdot v_2) = e(u, v_1) \cdot e(u, v_2)$$

(2) Non-degeneracy. For $u, v \in G$ and $u \neq v$, e is anti-symmetrical: $e(u, v) \neq 1$.

(3) Bilinearity. For all $u, v \in G$ and $a, b \in Z_P$:

$$e(u^a, v^b) = e(u, v)^{a \cdot b}$$

(4) Computability. There exists an efficiently computable algorithm for computing e.

Table 1 shows some notations and their descriptions.

Table 1. Notations and Descriptions

Symbol	Mathematical Formulation	Physical Meaning
$\pi_{key}()$	$\{0,1\}^{log_2(n)} \times K \rightarrow \{0,1\}^{log_2(n)}$	a pseudorandom permutation
$f_{key}()$	$\{0,1\}^* \times K \rightarrow Z_P$	a pseudorandom function
$H(\cdot)$	$\{0,1\}^* \rightarrow G$	a secure map-to-point hash function
$h(\cdot)$	$G \rightarrow Z_P$	a secure cryptographic hash function

3. Review the Worku et al.'s scheme

For ease of description, we omit the batch auditing and any other inessential details.

The data file $F = \{m_i\}_{i \in [1,n]}$ is stored in the cloud server. Worku et al.'s scheme consists of four basic algorithms: *KeyGen*, *SigGen*, *ProofGen* and *VerifyProof*.

$(pk, sk) \leftarrow KeyGen(1^k)$: The user generates a random signing key pair *(ssk , spk)*, then he randomly chooses *x*, $u \in G$, and computes $v = g^x \in G$. The user then stores $sk = \{x, ssk\}$ as his secret parameters and states $pk = \{u, v, g, spk\}$ as public parameters.

$(\phi) \leftarrow SigGen(sk, F)$: The user chooses a random element *name* for file naming and computes the file tag as $t = name \,||\, Sig_{ssk}(name)$, and generates a signature σ_i for each block m_i as follows:

$$\sigma_i = (H(i) \cdot u^{m_i})^x \in G \; (1 \leq i \leq n)$$

Then he sends $\{F, \phi = \{\sigma_i\}_{1 \leq i \leq n}, t\}$ to the cloud server for storage. Any time when the TPA wants to start the auditing protocol, it first retrieves the tag *t* and checks its validity by *spk*. Then it constructs a challenge $chal = \{c, k_1, k_2\}$, where *c* is the number of data blocks to be checked and k_1, k_2 are pseudorandom permutation keys chosen randomly by the TPA for each auditing.

$(P) \leftarrow ProofGen(F, \phi, chal)$: Upon receiving the *chal*, the cloud server determines the subset $I = \{s_j\}(1 \leq j \leq n)$ of set $[1, n]$, and computes $s_j = \pi_{k_1}(j)$, $v_{s_j} = f_{k_2}(j)(1 \leq j \leq c)$ and $r = f_{k_3}(chal)$, where k_3 is a pseudorandom function key generated by the cloud server

for each auditing. And then, the server computes: $R = u^r \in G$, $\mu^* = \sum_{i \in I} v_i \cdot m_i$, $\mu = \mu^* + r \cdot h(R)$ and $\sigma = \prod_{i \in I} \sigma_i^{v_i}$. Finally, the cloud server sends (μ, σ, R) to the TPA.

$(True, False) \leftarrow VerifyProof(pk, chal, P)$: After receiving the proof from the cloud server, the TPA computes $s_j = \pi_{k_1}(j)$ and $v_{s_j} = f_{k_2}(j)$, where $1 \le j \le c$ and verifies the proof by the following equation:

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

If the equation holds, output "True"; Otherwise, output "False".

4. Cryptanalysis of the Worku et al.'s scheme

The Worku et al.'s scheme owns privacy-preserving guarantee and can be extended to support batch auditing. And the authors claim that their scheme is provably secure in the random oracle model. Although a formal proof is given in [3] to prove the scheme is secure, there still exists a strong adversary in the real cloud application scenario. For example, an adversary, who is on-line and active, can modify the outsourced data in the way he needs and also modify an interaction messages between cloud server and TPA in the network, in order to fool the TPA and the cloud user to trust that the data are well maintained by the cloud server.

Assume the adversary modifies each data block m_i to $m_i^* = m_i + l_i$ for $i \in [1, n]$ and he records how the user's data are modified. In the *ProofGen* phase, cloud server computes $\hat{\mu}*$ and $\hat{\mu}$ as:

$$\hat{\mu}* = \sum_{i \in s_c} v_i \cdot m_i^*$$

$$\hat{\mu} = \hat{\mu}* + r \cdot h(R)$$

Then the cloud server sends $\{\hat{\mu}, \sigma, R\}$ to the TPA. The adversary intercepts this invalid proof, computes v_i and modifies it to $\{(\hat{\mu} - \sum_{i \in s_c} v_i l_i), \sigma, R\}$. Then the adversary sends the modified proof to the TPA.

After receiving the proof, the TPA verifies the following equation:

$$e(\sigma, g) = e(\prod_{i \in I} H(i)^{v_i} \cdot u^{\hat{\mu} - \sum_{i \in s_c} v_i l_i} \cdot R^{-h(R)}, v)$$

For the above equation, the right-hand side as:

$$e(\prod_{i \in I} H(i)^{v_i} \cdot u^{\hat{\mu} - \sum_{i \in s_c} v_i l_i} \cdot R^{-h(R)}, v)$$

$$= e(\prod_{i \in I} H(i)^{v_i} \cdot u^{(\hat{\mu}* + rh(R)) - \sum_{i \in I} v_i l_i} \cdot R^{-h(R)}, v)$$

$$= e(\prod_{i \in I} H(i)^{v_i} \cdot u^{(\sum_{i \in I} v_i (m_i + l_i) + rh(R)) - \sum_{i \in I} v_i l_i} \cdot R^{-h(R)}, v)$$

$$= e(\prod_{i\in I} H(i)^{v_i} \cdot u^{\sum_{i\in I} v_i m_i + rh(R)} \cdot u^{-rh(R)}, g^x)$$

$$= e(\prod_{i\in I} H(i)^{v_i} \cdot u^{\sum_{i\in I} v_i m_i}, g)^x$$

$$= e(\prod_{i\in I} (H(i)\cdot u^{m_i})^{xv_i}, g)$$

$$= e(\prod_{i\in I} \sigma_i^{v_i}, g)$$

$$= e(\sigma, g)$$

$$= \textbf{the left-hand side}$$

Thus, the verification equation holds. In this way, the TPA seemingly has every reason to believe that the data stored in cloud server are well maintained. Actually, it is not true. Therefore, the adversary successfully modifies the outsourced data while passing the verification.

The worku et al.'s scheme owns the security flaw, since the adversary can modify the forge proof to the valid proof. Essentially, the cloud server uses a random mask code to blind the user's information for privacy-preserving, but there exists definite linear relationship between the random mask code and the blinded information. This definite linear relationship causes the security flaw which exists in original scheme mentioned before.

5. The proposed scheme

In this section, we propose a secure and efficient public auditing scheme. The scheme fixes the aforementioned security flaw while retaining all the features of the Worku et al.'s scheme. Our scheme employs a nonlinear disturbance code to change the definite linear relationship between the random mask code and the blinded information to non-linear relationship.

The proposed scheme has four basic algorithms (*KenGen*, *SigGen*, *ProofGen* and *VerifyProof*). Same as the Worku et al.'s scheme, we assume the data file $F = \{m_i\}_{i\in[1,n]}$ is stored in the cloud server.

In the *KenGen* algorithm, a cloud user generates a random signing key pair *(ssk , spk)*, randomly chooses *x*, $u \in G$ and computes $v = g^x \in G$. Here, the user's secret parameters are $sk = \{x, ssk\}$ and public parameters are $pk = \{u, v, g, spk\}$.

In the *SigGen* algorithm, the user chooses a random element *name* for file naming and computes the file tag as $t = name \,||\, Sig_{ssk}(name)$, and generates a signature σ_i for each block m_i as follows:

$$\sigma_i = (H(i \,||\, name)\cdot u^{m_i})^x \in G \ (1 \le i \le n)$$

And then, the user sends $\{F, \phi = \{\sigma_i\}_{1\le i\le n}, t\}$ to the cloud server for storage. Whenever the TPA wants to check the integrity of the cloud-stored data, it first retrieves the tag *t* and checks its validity by *spk*. Then it constructs a challenge $chal = \{c, k_1, k_2\}$, where *c* is the number of data blocks to be checked and k_1, k_2 are pseudorandom permutation keys chosen randomly by the TPA for each checking. Finally, the TPA sends *chal* to the cloud server.

In the *ProofGen* algorithm, the cloud server determines the subset $I=\{s_j\}(1\le j\le n)$ of set $[1,\ n]$, computes $s_j=\pi_{k_1}(j)$, $v_{s_j}=f_{k_2}(j)(1\le j\le c)$ and $r=f_{k_3}(chal)$, where k_3 is a pseudorandom function key generated by the cloud server for each checking. And then, the server computes: $R=u^r\in G$, $\mu^*=\sum_{i\in I}v_i\cdot m_i$, $\mu=r^{-1}\cdot(\mu^*+h(R))$ and $\sigma=\prod_{i\in I}\sigma_i^{v_i}$. Finally, the cloud server sends (μ,σ,R) to the TPA.

In the *VerifyProof* () algorithm, the TPA verifies the following equation:

$$e(\sigma,g)=e(\prod_{i\in I}H(i\,||\,name)^{v_i}\cdot R^{\mu}\cdot u^{-h(R)},v)\rightarrow(1)$$

If the equation holds, output "True"; Otherwise, output "False".

5.1 Support for batch auditing

Our scheme also supports the batch auditing.

If there are K different users with K different data, let $U=\{U_x\}(x\in K)$ be the set containing all these users. Each user U_x has a data file $F_x=\{m_{x,i}\}_{1\le i\le n}$ to be outsourced to the cloud server. Firstly, each user U_x generates his secret parameters (α_x,ssk_x) and public parameters $(u_x,v_x=g^{\alpha_x},g,spk_x)$ independently. For U_x $(x\in K)$, he chooses a random element $name_x$, as the identifier of the data file F_x. Then U_x calculates his file tag $t_x=name_x\,||\,Sig_{ssk_x}(name_x)$. Choosing u_x from G randomly, and each of them computes a signature for every $m_{x,i}$ $(x\in[1,K],i\in[1,n])$ as:

$$\sigma_{x,i}=(H(x\,||\,i\,||\,name)\cdot u_x^{m_{x,i}})^{\alpha_x}$$

Finally, all users send $\{F_x,\phi=\{\sigma_{x,i}\}_{1\le i\le n},t_x\}\,(x\in[1,K])$ to the cloud server for storage.

Any time when the TPA wants to start the auditing protocol, it makes some necessary calculations and get the challenge parameters $chal=\{c,k_1,k_2\}$ for auditing. Then the TPA sends the $chal$ to the cloud server.

After receiving $chal$, the cloud server determines the subset $I=\{s_1,s_2,...,s_k\}$. It randomly chooses $r_x\in Z_p$, computes r_x^{-1} and $R_x=u_x^{r_x}$ for each user. And then, the cloud server computes:

$$\mu_x=r_x^{-1}\cdot(\sum_{i\in I}v_i m_{x,i}+h(R_x))$$

$$\sigma=\prod_{x\in K}(\prod_{i\in I}\sigma_{x,i}^{v_i})$$

The cloud server then sends the proof $P=\{\sigma,\{\mu_x\}_{x\in I},\{R_x\}_{x\in I}\}$ to the TPA.

After receiving the proof from the cloud server, the TPA verifies the data integrity by the following equation:

$$e(\sigma,g)=\prod_{x\in K}e(\prod_{i\in I}H(x\,||\,i\,||\,name)^{v_i}\cdot R_x^{\mu_x}\cdot u_x^{-h(R_x)},v_x)\rightarrow(2)$$

6. Evaluation

In this section, we give an overall evaluation in the proposed scheme. It consists of correctness proof, security analysis, performance analysis.

6.1 Correctness Proof

Here, we give the correctness proof of the verifiable equation (1) and (2). It guarantees that our scheme is credible.

For (1), the left-hand side as:

$$e(\sigma, g) = e(\prod_{i \in I} (\sigma_i^{v_i}), g)$$

$$= e(\prod_{i \in I} (H(i \| name) \cdot u^{m_i})^{x \cdot v_i}, g)$$

$$= e(\prod_{i \in I} H(i \| name)^{v_i} \cdot u^{\sum_{i \in I} m_i \cdot v_i}, g^x)$$

$$= e(\prod_{i \in I} H(i \| name)^{v_i} \cdot u^{\mu^*}, v)$$

$$= e(\prod_{i \in I} H(i \| name)^{v_i} \cdot u^{\mu \cdot r - h(R)}, v)$$

$$= e(\prod_{i \in I} H(i \| name)^{v_i} \cdot R^{\mu} \cdot u^{-h(R)}, v)$$

$$= \textbf{The right-hand side}$$

For (2), the left-hand side as:

$$e(\sigma, g) = e(\prod_{x \in K} \prod_{i \in I} \sigma_{x,i}^{v_i}, g)$$

$$= \prod_{x \in K} e(\prod_{i \in I} (H(k \| i \| name) \cdot u_x^{m_{x,i}})^{v_i}, g^{\alpha_x})$$

$$= \prod_{x \in K} e(\prod_{i \in I} H(k \| i \| name)^{v_i} \cdot u_x^{\sum_{i \in I} m_{x,i} \cdot v_i}, v_x)$$

$$= \prod_{x \in K} e(\prod_{i \in I} H(k \| i \| name)^{v_i} \cdot u_x^{\mu_x \cdot r_x - h(R)}, v_x)$$

$$= \prod_{x \in K} e(\prod_{i \in I} H(k \| i \| name)^{v_i} \cdot R_x^{\mu_x} \cdot u_x^{-h(R_x)}, v_x)$$

$$= \textbf{The right-hand side}$$

6.2 Security analysis

Here, we first prove that the proposed scheme fixes the security flaw which exists in the original scheme. Then, we give a formal proof of the proposed scheme's storage security. Finally, we prove the scheme can preserve the cloud user's privacy. Our security analysis depends on the hardness assumption of discrete logarithm problem (DLP) and the hardness assumption of Computational Diffie-Hellman problem (CDH). The **Definition 1** recalls DLP

on G. And the **Definition 2** recalls CDH on G.

Definition 1: DLP states that given $h, g \in G$ as input, compute $a \in Z_p$ such that $h = g^a$.

Definition 2: CDH problem states that given g, g^a, $g^b \in G$, where a, $b \in Z_p$, as input, compute $h = g^{ab}$.

6.2.1 Fixing the security flaw existing in the Worku et al.' scheme

We suppose that there exists an adversary modifying each data m_i to $m_i* = m_i + l_i$ for $i \in [1, n]$. The adversary records how the cloud user's data are modified. In the auditing process, the TPA and the cloud server honestly execute the protocol. That is, in the $SigGen()$ phase, the TPA sends a challenge $chal = \{c, k_1, k_2\}$ to the cloud server. In the $ProofGen()$ phase, after calculating $s_i, v_{s_i}, r, R, \sigma$ and r^{-1}, the cloud server computes:

$$\hat{\mu}* = \sum_{i \in I} (m_i + l_i) \cdot v_i$$

$$\hat{\mu} = r^{-1} \cdot (\hat{\mu}* + h(R))$$

$$= r^{-1} \cdot (\mu^* + \sum_{i \in I} l_i v_i + h(R))$$

$$= r^{-1} \cdot (\mu^*) + r^{-1} \cdot \sum_{i \in I} l_i v_i + r^{-1} h(R)$$

Then the cloud server sends $Proof = \{\hat{\mu}, R, \sigma\}$ to the TPA. The adversary intercepts $Proof$ on the channel. However, if the adversary attempts to modify the $Proof$ to the valid $Proof$, he must modify the $\hat{\mu}$ to μ. That is, he should compute $\hat{\mu} - r^{-1} \cdot \sum_{i \in I} l_i v_i$. We notice that r is randomly chosen by the cloud server and is unknown to the adversary, and $R = u^r \in G$, due to the hardness assumption of DLP, the adversary is still agnostic of the values r and r^{-1}. Therefore, our scheme can resist the aforementioned attack.

6.2.2 Storage security assurance

We need to prove that cloud server cannot generate valid proof P without storing the correctness and integrity data, as captured by **Theorem 1**.

Theorem 1: If the cloud server passes the phase of data auditing, it must possess truly the specified data intact.

Proof. As [3], there exists a challenger controlling the random oracle $H(\cdot)$, the malicious cloud server is treated as an adversary. If the adversary can forge a valid auditing proof to pass the verification with non-negligible probability, the challenger can construct a simulator that can solve the CDH problem.

The simulator randomly chooses a, b from Z_p and h from G. Set $v = g^{\alpha}$, $u = g^a h^b$.

For each i in the challenge, the simulator chooses r_i from Z_p and the file identifier $name$, and processes the random oracle:

$$H(i \| name) = \frac{g^{r_i}}{g^{a \cdot m_i} \cdot h^{b \cdot m_i}} \rightarrow (3)$$

We note that $u = g^a h^b$ and

$$H(i \| name) \cdot u^{m_i} = \frac{g^{r_i}}{g^{a \cdot m_i} \cdot h^{b \cdot m_i}} \cdot g^{a \cdot m_i} \cdot h^{b \cdot m_i} = g^{r_i}$$

Therefore, the simulator calculates $\sigma_i = (H(i \| name) \cdot u^{m_i})^{\alpha} = (g^{\alpha})^{r_i}$ for signature query.

Actually, for the challenger, $P = \{\sigma, \mu, R\}$ is the valid response from the cloud server. And in this case, given $(g, g^{\alpha}, R) \in G$, the simulator wants to output R^{α}. Here, we stress in particular that it is difference from the original CDH problem. However, the adversary have been computed the r and hidden from the challenger, and then, R is definite and public. Thus, the simulator outputs R^{α} is also a CDH problem [11].

Now, we will go on proving the **Theorem 1**. The aforementioned P can meet verification equation (1).

$$e(\sigma, g) = e(\prod_{i \in I} H(i \| name)^{v_i} \cdot R^{\mu} \cdot u^{-h(R)}, v)$$

However, the malicious cloud server will try to forge the response proof as $P' = \{\sigma', \mu', R\}$ while r is the same as before. Thus, the response P' can also meet the equation as follows:

$$e(\sigma', g) = e(\prod_{i \in I} H(i \| name)^{v_i} \cdot R^{\mu'} \cdot u^{-h(R)}, v)$$

As the challenger's process defined in the security model of original scheme, if $\sigma = \sigma'$, the challenger stop responding to the adversary. So $\sigma \neq \sigma'$ and $\mu \neq \mu'$. Here, we define $\Delta\mu = \mu' - \mu$, $\Delta\mu^* = \mu^{*'} - \mu^*$, and the adversary can solve the CDH problem as follows:

$$e(\frac{\sigma'}{\sigma}, g) = e(\prod_{i \in I} (\frac{\sigma_i'}{\sigma_i})^{v_i}, g) = e(\prod_{i \in I} (\frac{u^{m_i' v_i}}{u^{m_i v_i}})^{\alpha}, g) = e(u^{\sum_{i \in I} (m_i' v_i - m_i v_i)}, v) = e(u^{\mu^{*'} - \mu^*}, v) = e(u^{\Delta\mu^*}, v) \rightarrow (4)$$

Because r is same for the two verification equations above, and $\mu = r^{-1} \cdot (\mu + h(R))$, we can get $\Delta\mu \cdot r = \Delta\mu^*$, and further get:

$$e(\frac{\sigma'}{\sigma}, g) = e(u^{\Delta\mu \cdot r}, v) \rightarrow (5)$$

And because $R = u^r, v = g^{\alpha}$, according to the bilinear property, we can rearrange and simplify the equation (5) as follows:

$$e((\sigma' \sigma^{-1})^{\frac{1}{\Delta\mu}}, g) = e(R^{\alpha}, g)$$

It is clear that $R^{\alpha} = (\sigma' \sigma^{-1})^{\frac{1}{\Delta\mu}}$. Our premise is that $\mu' \neq \mu$, thus $\Delta\mu \neq 0$. Therefore, we can compute R^{α}. However, it contradicts to the hardness assumption of CDH. Therefore, $\sigma' = \sigma$.

The simulator can solve the DLP, only if the adversary success probability is non-negligible.

As described before, since $\sigma' = \sigma$, μ' is different from μ. The challenger answers the queries from the adversary and we have:

$$e(\sigma, g) = e(\sigma', g)$$

$$e(\prod_{i \in I} H(i \| name)^{v_i} \cdot R^{\mu} \cdot u^{-h(R)}, v) = e(\prod_{i \in I} H(i \| name)^{v_i} \cdot R^{\mu'} \cdot u^{-h(R)}, v)$$

$$e(R^{\mu}, v) = e(R^{\mu'}, v)$$

$$u^{r\Delta\mu} = 1$$

It means $r\Delta\mu = 0 \mod p$, and because the r is uniform, we can deduce $\mu' = \mu$. But it is inconsistent with our assumption. Therefore, $u^{r\Delta\mu} = 1$. In this case, we can solve the DLP as follows:

$$1 = u^{r\Delta\mu} = (g^a h^b)^{r\Delta\mu} = g^{ra\Delta\mu} \cdot h^{rb\Delta\mu}$$

The solution to DLP is:

$$h = g^{-\frac{ra\Delta\mu}{rb\Delta\mu}} = g^{-\frac{a}{b}}$$

However, the probability of $b = 0$ just only $1/p$, and can be ignored. This completes the proof of the **Theorem 1**.

6.2.3 Privacy-preserving Assurance

The following theorem indicates that the TPA cannot recover users' data during the verification process. Concretely, the TPA cannot recover μ^* from the security perspective.

Theorem 2. The TPA cannot recover μ^* from the cloud server's $Proof = \{\mu, \sigma, R\}$.

Proof. While the TPA tries to recover the user's data, it controls c in $chal$ and obtains enough linear combinations of the data block m_i and its corresponding element v_i, and then, it achieves the goal by solving this system of linear equations. On this occasion, when the cloud server generates a valid proof, it blinds μ^* using r^{-1} which is the inverse element of the random mask r. If the TPA still attempts to get μ^*, there are two methods to attain. One is to immediately obtains r^{-1}, the other is to compute r^{-1} by R. Since $r = f_{k_3}(chal)$, k_3 is randomly chosen by the cloud server and is unknown to the TPA, the TPA cannot work out r and r^{-1}. Therefore, the former method is not workable. For the latter method, note that $R = u^r \in G$, due to the hardness assumption of DLP, the value r is still unknown to the TPA. As a consequence, μ assures the privacy of μ^*.

This completes the proof of the **Theorem 2**.

6.3 Performance Analysis

We give elaborate performance analysis in order to show the efficiency of the proposed scheme. In our experiment, the process of the user, the server and the TPA are implemented on a windows 7 system with an Intel Core 2 i5 CPU running at 2.53 GHz, 2 GB DDR 3 of RAM(1.74 GB available). All algorithms are implemented by C language, and our code uses the MIRACL library version 5.6.1. The elliptic curve we used is a MNT curve, where the base

field size is 159 bits and the embedding degree is 6. The security level is chosen to be 80 bits, it means that $|v_i|=80$ and $|p|=160$. All the results of experiment are represented the average of 30 trials.

In the following, we emphasize on reporting our performance results from computational overhead. And we also give a performance comparison with [3]. According to the comparison, we can see that our scheme retains the efficiency of [3] while fixing its security flaw.

6.3.1 The Proposed Scheme's Computation Overhead

Firstly, We specify some notations represent the computation of corresponding operation (refer to Table 2).

Table 2. Notation of Operations

Symbol	Corresponding Operation
$Exp_{\langle p\rangle}$	exponentiation $g^x mod p$, for $g \in G, x \in Z_p$ and p is a prime value
$Hash_{Z_p}$	hash a value into Z_P
$Hash_G$	hash a value into G
$Mult_G$	multiplication in group G
$Pair_{G_T}$	computing pairing $pair = e(u,v)$ where $u, v \in G$ and $pair \in G_T$
Add_{Z_p}	addition in Z_P
$ComInver_{Z_p}$	computing inverse element in Z_P

For our scheme, on the cloud user side, the main calculation is in computing public parameters, the file tag and the data blocks' signatures (we use DSS to sign the data need to be signed). His computation cost is:

$$(2n+2)\cdot Exp_{\langle p\rangle} + (n+2)\cdot Mult_G + Add_{Z_p} + Hash_G + Hash_{Z_p}$$

On the TPA side, before generating $chal$, it retrieves the file tag T. After receiving $Proof$, the TPA checks the verification equation. The corresponding computation cost is $3\cdot Mult_G + 2\cdot Exp_{\langle p\rangle} + Hash_{z_p}$ and $2\cdot Pair_{G_T} + (c+2)\cdot Exp_{\langle p\rangle} + Hash_G + 2\cdot Mult_G$.

Similarly, on the cloud server side, it computes μ^*, θ, R and μ^*. The corresponding computation cost is :

$$(c+1)\cdot Mult_G + (c+1)\cdot Exp_{\langle p\rangle} + (c+1)\cdot Add_{Z_p} + Hash_{Z_p} + ComInver_{Z_p}$$

Compared with [3], our scheme just additionally calculates an inverse element in cloud server side. The operation of computing inverse element is too small to be ignored. Therefore, in a practical system, our scheme is of high efficiency as [3]. And **Fig. 2** shows the performance of TPA in different challenge blocks c.

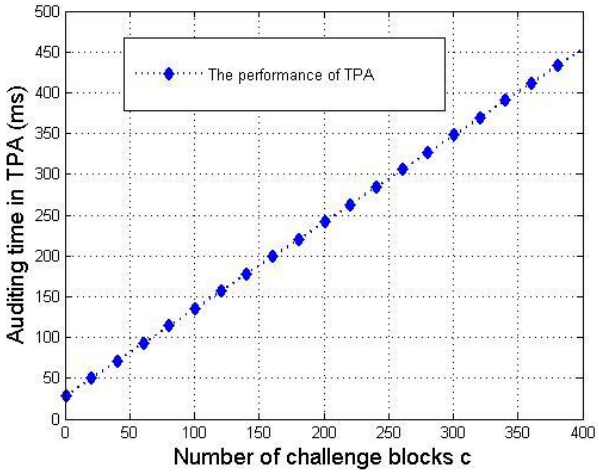


Fig. 2. The performance of TPA in different challenge blocks

6.3.2 Batch Auditing Overhead with Its Advantage

Fig. 3 and **Fig. 4** respectively indicate the efficiency comparison on auditing time between batch auditing and basic auditing in $c = 300$ and $c = 500$. The experiment shows that batch auditing improves efficiency a lot.

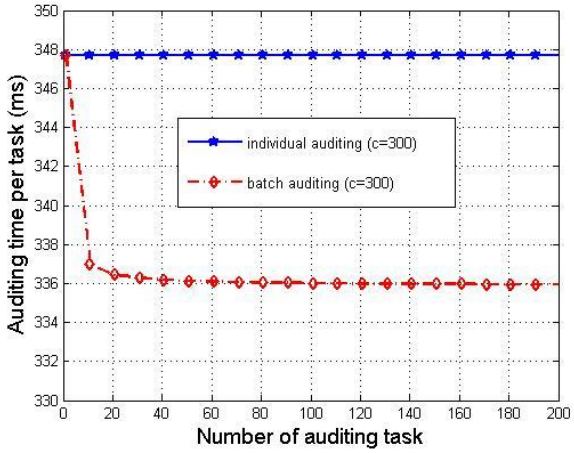


Fig. 3. Comparison on auditing time between batch and individual auditing (c=300)

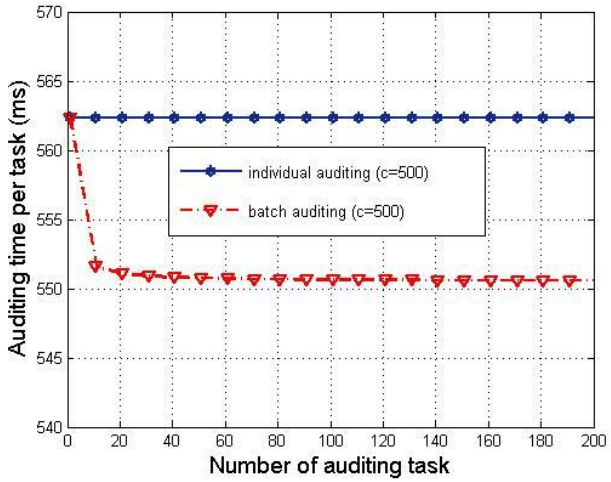


Fig. 4. Comparison on auditing time between batch and individual auditing (c=500)

Through above fomal security proof and performance analysis, we can see that our scheme fixes the aforementioned security flaw while ensuring the same efficiency. Therefore, the proposed scheme has advantages over the [3] in a practical application.

7. Conclusion

In this paper, we give a cryptanalysis in Worku et al.'s scheme, and prove their scheme has a security flaw. Exploiting the security flaw, an adversary is able to arbitrarily modify the cloud user's data while avoiding the detection. Furthermore, we propose an efficient and provable secure public auditing scheme for cloud storage. The proposed scheme fixes the security flaw existing in the Worku et al.'s scheme while retaining all features. The formal security proof and performance analysis demonstrate that our scheme is secure and as efficient as worku et al.'s scheme.

8. Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.61370203) and the Science and Technology on Communication Security Laboratory Foundation (Grant No. 9140C110301110C1103)

References

[1] Wang, C, Wang, Q, Ren, K and Lou, W, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proc. of IEEE Conf. on Computer Communication*, pp, 1-9, March 14-19, 2010. Article (CrossRef Link)

[2] Yang K and Jia X, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," *IEEE Transactions on Parallel and Distributed Systems,* vol. 24, no. 9, pp. 1717-1726, September, 2013. Article (CrossRef Link)

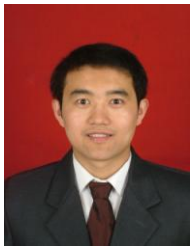
[3] Worku S G, Xu C, Zhao J and He X, "Secure and efficient privacy-preserving public auditing scheme for cloud storage," *Computers & Electrical Engineering*, vol. 40, no. 5, pp. 1703−1713, July, 2014. Article (CrossRef Link)

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

2013. Article (CrossRef Link)
[5] Zhao J, Xu C, Li F and Zhang W, "Identity-Based public verification with privacy-preserving for data storage security in cloud computing," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences,* vol. 96, no. 12, pp. 2709-2716, December, 2013. Article (CrossRef Link)
[6] Xu C, He X and Abraha-Weldemariam D, "Cryptanalysis of Wang's auditing protocol for data storage security in cloud computing," in *Proc. of International Conf. on Information Computer Application,* pp. 422-428, September 14-16, 2012. Article (CrossRef Link)
[7] Ni, J, Yu, Y, Mu, Y and Xia, Q, "On the Security of an Efficient Dynamic Auditing Protocol in Cloud Storage," *IEEE Transactions on Parallel and Distributed Systems,* vol. 25, no. 10, pp. 2760 – 2761, October, 2013. Article (CrossRef Link)
[8] Li, H, Lin, X, Yang, H, Liang, X, Lu, R, and Shen, X, "EPPDR: An Efficient Privacy-Preserving Demand Response Scheme with Adaptive Key Evolution in Smart Grid," *IEEE Transactions on Parallel and Distributed Systems,* vol. 25, no. 8, pp. 2053 - 2064, August, 2014. Article (CrossRef Link)
[9] Giuseppe A, Randal B, Reza C, Joseph H, Lea K, Zachary P and Dawn S, "Provable data possession at untrusted stores," in *Proc. of ACM conf. on computer and communications security,* pp. 598-609, October 29–November 2, 2007. Article (CrossRef Link)
[10] Shacham H and Waters B, "Compact proofs of retrievability," in *Pro.of International Conf. on the Theory and Application of Cryptology and Information Security*, pp. 90-107, December 7-11, 2008. Article (CrossRef Link)
[11] Boldyreva A, Gentry C, O'Neill A and Yum, D. H, "Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing," in *Proc. of ACM conf. on Computer and communications security*, pp. 276-285, October 29–November 2, 2007. Article (CrossRef Link)

Chunxiang Xu received her B.Sc., M.Sc. and Ph.D. degrees at Xidian University, in 1985, 1988 and 2004 respectively, P.R.China. She is presently engaged in information security, cloud computing security and cryptography as a professor at University of Electronic Science Technology of China (UESTC).

Yuan Zhang received his B.Sc. degree in University of Electronic Science Technology of China (UESTC) in 2013,P.R.China. He is currently a master student in School of Computer Science and Engineering at University of Electronic Science Technology of China. His research interests are cryptography, network security and Cloud Computing security.



Yong Yu received his Ph.D. degree in cryptography from Xidian University in 2008. He is currently an associate professor of University of Electronic Science and Technology of China and a Vice Chancellor's research fellow of the University of Wollongong as well. His research focuses on cryptography and its applications, especially public encryption, digital signature and secure cloud storage.

Xiaojun Zhang received his B.Sc. degree in mathematics and applied mathematics at Hebei Normal University in 2009,P.R.China and received M.Sc degree at Guangxi University in 2012. He is a Ph.D. degree candidate in information security at University of Electronic Science Technology of China (UESTC). He is a student member of CACR. He is presently engaged in cryptography, network security and cloud computing security.

Junwei Wen received his B.Eng. degree from Southwest Jiaotong University(SWJTU) in 2009,P.R.China. He is currently a master student in School of Computer Science and Engineering at University of Electronic Science Technology of China (UESTC). His research interests is in intrusion detection, network security.