Hindawi
Journal of Healthcare Engineering
Volume 2018, Article ID 9638680, 9 pages
https://doi.org/10.1155/2018/9638680

Research Article

Genuine and Secure Identity-Based Public Audit for the Stored Data in Healthcare Cloud

Jianhong Zhang,[1,2] **Zhibin Sun**,[1] **and Jian Mao**[3]

[1]*School of Electronic and Information Engineering, North China University of Technology, Beijing 100144, China*
[2]*Guangxi Key Laboratory of Cryptography and Information Security, Guilin 541004, China*
[3]*School of Electronic and Information Engineering, Beihang University, Beijing 100191, China*

Correspondence should be addressed to Jianhong Zhang; zjhncut@163.com

Received 31 July 2017; Revised 14 December 2017; Accepted 22 July 2018; Published 18 September 2018

Academic Editor: Zong-Min Wang

Cloud storage has attracted more and more concern since it permits cloud users to save and employ the corresponding outsourced files at arbitrary time, with arbitrary facility and from arbitrary place. To make sure data integrality, numerous public auditing constructions have been presented. However, existing constructions mainly have built on the PKI. In these constructions, to achieve data integrality, the auditor first must authenticate the legality of PKC, which leads to a great burden for the auditor. To eliminate the verification of time-consuming certificate, in this work, we present an efficient identity-based public auditing proposal. Our construction is an identity-based data auditing system in the true sense in that the algorithm to calculate authentication signature is an identity-based signature algorithm. By extensive security evaluation and experimental testing, the consequences demonstrate that our proposal is safe and effective; it can efficiently hold back forgery attack and replay attack. Finally, compared with the two identity-based public auditing proposals, our proposal outperforms the two proposals under the condition of overall considering computational cost, communication overhead, and security strength.

1. Introduction

With the technique progress in communication filed, the amount of the generated data is going through fast growth. Many companies working on the healthcare trade increasingly make use of cloud storage services. Instead of every hospital storing and maintaining medical data in physical servers, cloud storage is becoming a popular alternative since it can offer the clients more convenient network-connection service, on-demand data storage service, and resource-sharing service.

The aging population problem urges healthcare services to make the continuous reformation so as to obtain cost-effectiveness and timeliness and furnish the services of higher quality. Numerous specialists deem that cloud-computing technique may make healthcare services good by reducing EHC (electronic-health-record) start-up costs, such as software, equipment, employee, and various license fees. These reasons will urge to adopt the relevant cloud techniques. Let us see one instance of healthcare services in which cloud technique is applied, the *Healthcare Sensor* system can automatically collect the patients' vital data of the wearable devices which are connected to traditional medical equipment via wireless sensor networks and then upload these data to "medical cloud" for storage. Another typical instance is the *Sphere of Care by Aossia Healthcare*, it is started in 2015. These cloud-based systems can automatically collect every day real-time data of users. It alleviates manual collection burden so that the deployment of the whole medical system is simplified. However, they may make healthcare providers to face many challenges in migrating all local health data to the remote cloud server, where the paramount concerns are privacy and security since healthcare administrator no longer completely deals with the security of those medical records. After medical data are stored on the cloud, they are possibly corrupted or dropped.

To make sure the intactness of the stored data in the remote server, the patients or healthcare service providers

expect that the stored data integrality can be periodically checked to avoid the damage of the stored data. However, for the individuals, one of the greatest challenges is how to carry out the termly data integrality detection when the individuals have the copy of local files. Meanwhile, the method is also infeasible to conduct data integrality verification for a source-limited individual though retrieving the total data file.

To deal with the above issue, many specialists had presented a number of problem-solving methods which aim at the diverse systems and diverse security models in [1–20]. Nevertheless, most existing problem-solving methods had built on public key infrastructure (for short, PKI). As everyone knows that the PKI-based auditing proposals exist complex key management problem, data client needs to conduct key updating, key revocation, and key maintaining, and so on. Hence, the key management and certificate verification in the PKI-data auditing system will be a troublesome issue. Furthermore, PKC also needs more storage space than the individual ID since key pair (PK, sk) needs to be locally kept. For a verifier, to guarantee data integrality, it must firstly extract PKC from public key directory and then verify whether public key certificate (for short, PKC) is valid. Therefore, it also increases computation burden and communication overhead for the verifier.

In 2014, the first so-called ID-based data integrality proposal was proposed by Wang et al. [13]. Strictly speaking, their proposal is not a kind of identity-based auditing one because the algorithm to generate metadata authentication tag is not an idenity-based algorithm, but a PKC-based one. In 2015, Yu et al. put forward a generic method of constructing identity-based public auditing system by integrating the identity-based signature algorithm with traditional PDP protocols in [15]. Their research is very significant on studying the ID-based public auditing system. However, in their scheme, the algorithm to produce metablock authentication tag is still to adopt a PKI-based one. Furthermore, in the auditing phase, the auditor firstly verifies the validity of an identity-based signature on a public key PK, and then it executes data integrity verificationby using this public key PK again, which increases the computation burden of the auditor. In 2016, Zhang and Dong brought forward a novel identity-based public auditing proposal in [16]. The proposal is the identity-based public auditing system from the literal sense since their algorithm to produce metadata authentication tag is the ID-based signature algorithm. However, their scheme is shown to be insecure in Appendix.

To increase efficiency and strengthen the security of ID-based auditing protocols, in this work, a secure and efficient ID-based auditing construction is proposed. For our construction, its original contributions are as follows:

(1) On the basis of the idea of homomorphic signature in ID-based setting, we devise an authentic identity-based auditing proposal of data integrality. The proposal can not only avoid the key managing, but also relieve the auditor's burden.

(2) In auditing the phase, our scheme has constant communication overhead. Compared with the two schemes [15, 16], our proposal has more advantages with regard to computational cost and communication cost.

(3) In the random oracle model, the proposed proposal has serious security proof, and the corresponding proof can be tightly reduced to the CDH mathematic problem.

2. Architecture and Security of System

In the following chapter, in order to better understand our ID-based data integrality auditing protocol (ID-DIAP, for short), we firstly give a description of the system model, and afterwards the security model of our ID-DIAP for cloud storage is defined.

2.1. System Architecture. For our ID-DIAP system in cloud, the architecture is composed of four entities: privacy key generator (for short, the PKG), the third party verifier/auditor (for short, the TPA), cloud servers, and data user. The whole systematic architecture is demonstrated in Figure 1.

To avoid biases in the auditing process, the TPA is recommended to implement the audit function in our system model. Detail function of each role in system architecture is described as below.

(i) *Data User*. It acts as a cloud user and possesses a number of files which need to be uploaded to the remote cloud server without local data copy. Generally speaking, data user may be a resource-limited entity due to the limited capability of storing and computing. And it can flexibly access at any time and share the outsourced data.

(ii) *Cloud Servers*. They are composed of a group of distributed servers and have tremendous capability of storing and computing. Furthermore, it is answerable to save and maintain the stored files in cloud. Nevertheless, the cloud server might be untrusted, and for its own profits and a good commercial reputation, it might conceal data corruption incidents for its cloud users.

(iii) *The Third Auditor*. It acts as a verifier of data intactness. In principle, it has professional experience and practical capability to take charge data integrality audit in the person of cloud users/data users.

(iv) *The PKG*. It is a trusted entity and is duty bound to build up system parameters and to calculate privacy key of every cloud user.

For a cloud-based storing system, its goals are to alleviate the burden of data storage and maintaining of cloud users. Nevertheless, after data are uploaded to the remote sever in cloud, it might lead to a potential security problem since the uploaded data have been out of control for the data user and the remote server in cloud is generally unreliable. Data user might concern whether the stored files in cloud are intact. Thus, the data user wants some security measures to ensure

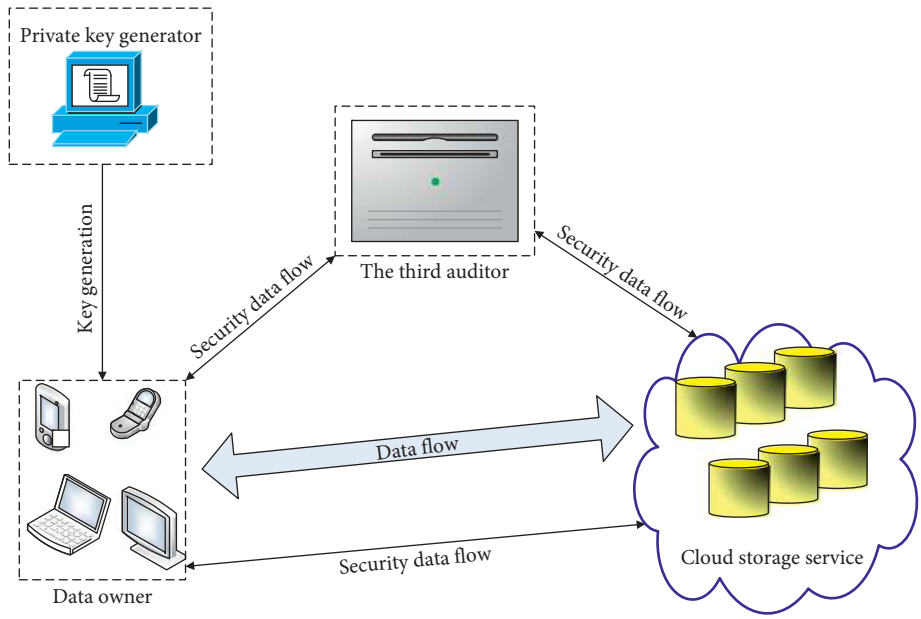

Figure 1: ID-based data storage model in cloud.

that the integrality of the outsourced data is examined regularly without a local copy.

Definition 1 (identity-based data integrity auditing protocol, ID-DIAP). In general, an ID-DIAP system contains the three stages:

(1) *System Initialization Phase*. In this phase, the PKG is duty bound to produce system parameters. Therefore, it runs *Setup*(1^k) algorithm to obtain system parameters *Para*, the PKG's key pair (*mpk*, *mpsk*) by inputting λ which is a safety parameter. On the contrary, the PKG also invokes KeyExtr(1^λ, Para, *mpsk*, ID) algorithm to calculate privacy key sk_{ID} for the data user with identity ID by inputting its *mpsk* and *Para* as well as the identity ID of the data user.

(2) *Data Outsourcing Phase*. In this phase, for the data owner (data user), it runs *TagGen*(M, $\mathrm{sk}_{\mathrm{ID}}$) to generate metadata authentication tag δ_i, on each data block m_i by inputting its private key sk_{ID} and the outsourced file M, where $M = m_1 \| \ldots \| m_n$. Finally, it uploads metadata authentication tags $\delta = \{\delta_1, \ldots, \delta_n\}$ to the cloud server.

(3) *Data Auditing Phase*: This phase is divided into three subphases: *Challenging*, *Proof*, and *Verifying*. Firstly, the auditor runs algorithm *Challenging*(M_{info}) to calculate *Chall* as the challenged information. After receiving *Chall*, the cloud server runs *Proof*(M, δ, and Chall) to calculate Prf as proving information, then returns Prf to the auditor. At last, the auditor invokes the algorithm *Verifying*(Chall, Prf, *mpk*, and M_{info}) to test whether the returned proving information Prf is valid.

2.2. Different Types of Attack and Security Definition. In the subsection, we will analyze that our ID-DIAP system may be confronted with diverse attacks in light of the behavior of every role in the system architecture. In our system architecture, the PKG is the privacy key generator which calculates data user's privacy key. In general, it is a credible authority. We assume that the PKG does not launch any security attack to the other entities in the whole system model. For the third auditor, it is deemed to be an honest-but-curious entity and can earnestly execute every step in the auditing course. And cloud server is considered to be unreliable. It might deliberately delete or alter rarely accessed data files for saving storage space. It is a powerful inside attacker in our security model. And the goal of the attacker is to tamper and replace the stored data without being found by the auditor. Because the cloud server is a powerful attacker in our security model, we mainly consider the attacks [7] which are launched by the cloud server in this paper.

2.2.1. Forge Attack. The vicious cloud server may produce a forged meta-authentication signature on a new data block or fabricate the fake proving information Prf to deceive the auditor by satisfying the auditing verification.

2.2.2. Replace Attack. If a certain data block in the challenge set was corrupted, the vicious cloud server would select another valid pair (m_i, δ_i) of data block and authentication tag to substitute the corrupted pair (m_j, δ_j) of data block and data tag.

2.2.3. Replay Attack. It is an efficient attack. With respect to the vicious storage server in cloud, it might produce the new proof information Prf* without retrieving the challenge data of the auditor though realizing the former proving information Prf.

3. Our Public Auditing Construction

In the following, we will give the description of our ID-DIAP system. It contains four entities: the PKG, cloud server, data user, and TPA. And the whole system is composed of five PPT algorithms. As for every entity and all algorithms, the diagram of the framework is represented in Figure 2. To clearly describe our protocol, the algorithms are given in detail below.

3.1. Setup. For the sake of enhancing readability, some notations used in our ID-DIAP system are listed in Table 1.

The PKG makes use of a parameter λ as an input and generates two cyclic groups G_1 and G_T. The two groups have the identical prime order $q > 2^k$. And let P and T be two generators of group G_1, where they satisfy $P \neq T$. And define a bilinear pairing map e: $G_1 \times G_1 \to G_T$. Next, it chooses two map-to-point cryptographic hash functions $H_0 : \{1,0\}^* \to G_1$ and $H_1 : \{1,0\backslash\}^* \to G_1$ and a resistant-collision hash function $H_2 : G_1 \times \{1,0\}^* \to Z_q$. And the PKG randomly chooses $s \in Z_q$ as its master privacy key, calculates $P_{\text{pub}} = sP$ as its public key. At last, public parameters Para are published as below:

$$\text{Para} = \left(G_1, G_T, q, e, P, T, H_0, H_1, H_2, P_{\text{pub}}\right). \quad (1)$$

And the PKG needs its master privacy key s to be secretly kept.

3.2. Key Extraction. For a data user, to produce its privacy key, it delivers its identification ID to the PKG. Subsequently, the PKG utilizes its master privacy key s and ID to implement the following process:

(1) Firstly, the data user delivers its identity information ID to the PKG.

(2) Next, the PKG generates $(D_{\text{ID0}}, D_{\text{ID1}})$ data user's privacy key, where

$$\begin{aligned} D_{\text{ID0}} &= sH_0(\text{ID}\|0), \\ D_{\text{ID1}} &= sH_0(\text{ID}\|1). \end{aligned} \quad (2)$$

And then it goes back $(D_{\text{ID0}}, D_{\text{ID1}})$ to the data user through a secret and secure channel.

(3) Upon receiving the private key $(D_{\text{ID0}}, D_{\text{ID1}})$, this data user is able to test whether its privacy key is valid through the following equations:

$$\begin{aligned} e\left(P_{\text{pub}}, H_0(\text{ID}\|0)\right) &= e\left(P, D_{\text{ID0}}\right), \\ e\left(P_{\text{pub}}, H_0(\text{ID}\|1)\right) &= e\left(P, D_{\text{ID1}}\right). \end{aligned} \quad (3)$$

3.3. TagGen Phase. For the upload data file M, firstly data file M is divided into n blocks by the data user, namely, $M = m_1\|m_2\|\ldots\|m_n$. To outsource this file M to the cloud, the data user needs to randomly choose a pair $(x_{\text{ID}}, Y_{\text{ID}})$ which is a private-public key pair of a secure signature algorithm $\sum = (\text{Sig}, \text{Ver})$, for example, BLS short signature.

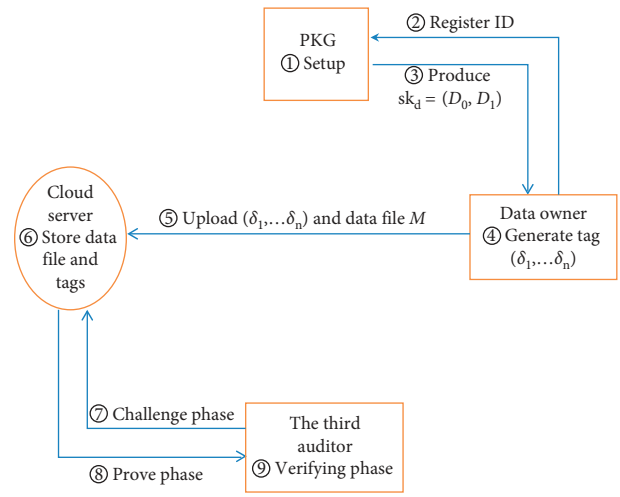

Figure 2: The relationship of the entities and algorithms.

Table 1: Notations in our ID-DIAP system.

Symbol	Meaning
K	A security parameter
H_0, H_1, H_2	Three hash functions
S	The master key of the PKG
T	A random generator of group G_1
Prf	The proof information
Chall	The challenge information
Q	A large prime number
G_T	A multiplicative group with the order q
G_1	An additive group with the order q
E	A bilinear pairing
$\|I\|$	The number of elements in set I

Let *Name* denote the identifier of data file M, and then it calculates the file authentication tag $\tau = \tau_0\|\sum\cdot\text{Sig}(x_{\text{ID}}, \tau_0)$, where $\sum\cdot\text{Sig}(x_{\text{ID}}, \tau_0)$ denotes a secure signature on τ_0, and τ_0 denotes an information string $\tau_0 =$ "Name$\|n$".

Subsequently, the data user needs to generate metadata authentication tag on the data block. To compute block authentication tags on all data blocks $\{m_i\}$ $i = 1, 2, \ldots, n$, the data user uniformly samples $r \in Z_q$ to calculate $R = rP$.

Next, for $i = 1$ to n, it calculates metadata authentication tag for data block m_i by the following steps:

(1) First of all, it calculates

$$h_i = H_2\left(\text{Index}_i \| \text{ID} \| R \| 0\right). \quad (4)$$

(2) And then, it makes use of its private key $(D_{\text{ID0}}, D_{\text{ID1}})$ to compute

$$\delta_i = rH_1\left(\text{Index}_i \| \text{Name}\right) + h_i D_{\text{ID0}} + m_i D_{\text{ID1}}. \quad (5)$$

(3) For data block m_i, the resultant authentication tag of the data block is $\theta_i = (R, \delta_i)$.

At last, the data user needs to upload all the meta-authentication tags $(\tau, R, \{\delta_i\}\ \{i = 1, 2, \ldots, n\})$ and the outsourced file M to the remote server in cloud.

On obtaining all the aforementioned data $(\tau, R, \{\delta_j\}\ \{j = 1, 2, \ldots, n\})$, the cloud server needs to execute the following validation procedure:

For $i = 1$ to n, it verifies the relation

$$e(\delta_i, P) = e\left(H_1\left(\text{Index}_i \| \text{Name}\right), R\right) e\left(h_i Q_0 + m_i Q_1, P_{\text{pub}}\right), \tag{6}$$

where $Q_1 = H_0(\text{ID}\|1)$ and $Q_0 = H_0(\text{ID}\|0)$. If all relations hold, then it parses τ into τ_0 and $\sum \cdot \text{Sig}(x_{\text{ID}}, \tau_0)$ and verifies the validity of signature

$$\sum \cdot \text{Ver}\left(\sum \cdot \text{Sig}(x_{\text{ID}}, \tau_0), \tau_0, Y_{\text{ID}}\right) = ? = 1. \tag{7}$$

If it is also valid, then the cloud server preserves these data in cloud.

3.4. Challenge Phase. To audit the integrality of the outsourced file M, firstly, the TPA parses τ into τ_0 and $\sum \cdot \text{Sig}(x_{\text{ID}}, \tau_0)$ and verifies $\sum \cdot \text{Ver}(\sum \cdot \text{Sig}(x_{\text{ID}}, \tau_0), \tau_0, Y_{\text{ID}}) = 1$. If it does not hold, then terminate it. Otherwise, it retrieves the corresponding file identifier name and block size n.

Later on, the auditor picks a subset $I \subseteq [1, n]$ randomly where $|I| = l$ and $\rho \in Z_q$ to generate a challenge information

$$\text{Chall} = \{\rho, I\}. \tag{8}$$

Finally, it delivers Chall to the cloud server as the challenge.

3.5. Proving Phase. After obtaining the corresponding challenge information $\text{Chall} = \{I, \rho\}$, for $i \in \{1, \ldots, |I|\}$, cloud server calculates $v_i = \rho^i \text{mod} q$, and then it produces a set $Q = \{i, v_i\}_{\{i \in I\}}$.

Subsequently, in light of the outsourced data file $M = \{m_1, \ldots, m_n\}$ and meta-authentication tag θ_i of each block, $i \in \{1, 2, \ldots, n\}$, it produces as follows:

$$\begin{aligned} \delta &= \sum_{j \in I} v_j \cdot \delta_j, \\ \mu &= \sum_{j \in I} v_j \cdot m_j. \end{aligned} \tag{9}$$

Finally, the cloud storage server goes back 3-tuple $\text{Prf} = (\delta, \mu, R)$ to the auditor as the corresponding proof information.

3.6. Verifying Phase. To check the outsourced data's integrality in cloud, after receiving the responded proof information $\text{Prf} = (\delta, \mu, R)$, the third auditor calculates as below:

$$\begin{aligned} \overline{h} &= \sum_{i \in I} v_i \cdot h_i, \\ \overline{H} &= \sum_{i \in I} v_i \cdot H_1\left(\text{Index}_i \| \text{Name}\right), \end{aligned} \tag{10}$$

where $h_i = H_2(\text{index}_i \| \text{ID} \| R \| 0)$ for $i \in I$.

Then, it checks the validity of the following equation:

$$e(\delta, P) = e(\overline{H}, R) e\left(\overline{h} Q_0 + \mu \cdot Q_1\right). \tag{11}$$

If the aforementioned Equation (11) satisfies, then the TPA outputs *VerifyRes* as true; if not, it outputs *VerifyReS* as false.

4. Security Analysis

To show our proposal's security, we will demonstrate that our proposal is proven to be secure against the above three attacks.

Theorem 1. *Assume there exists a PPT adversary Adv that is probabilistic polynomial-time attacker (for shot PPT) and can cheat the auditor using invalid proving information Prf which is forged by the adversary Adv (the dishonest cloud storage server) in a nonignorable probability ε, then we are able to design an algorithm B that can efficiently break the CDH assumption by invoking Adv as subprogram.*

Proof. Let us suppose that a PPT adversary {Adv} is capable to calculate a faked proving information Prf after the data blocks or metadata authentication tags are corrupted, then we are capable of constructing another a PPT algorithm B which is capable of breaking the CDH assumption by utilizing the adversary Adv. First of all, let a 3-tuple $(P, aP, bP) \in G_1$ be a CDH assumption's random instance, it is hard to obtain the solution abP.

To show the security proof, hash function H_0 in the game is regarded as random oracle, and identity ID of each data user is only made H_0-query once. For H_1 and H_2, they only act as one-way functions. In addition, the adversary Adv is capable of adaptively issuing the queries to three oracles: {H_0-oracle}, {Key-Extract oracle}, and {TagGen oracle}.

Setup. Choose two cyclic groups G_1 and G_T, and their orders are the same prime number q. The algorithm B firstly sets $P_{\text{pub}} = aP$ as the public key of the PKG. Let H_0, H_1, and H_2 be three hash functions. Finally, it sends public system parameters $(G_1, G_T, P, T, q, P_{\text{pub}}, e, H_0, H_1, H_2)$ to the adversary {Adv}. And let $j^* \in \{1, \ldots, qH_0\}$ be a challenged identity index of the data user.

H_0-Hash Oracle. The adversary {Adv} submits a query to H_0-oracle with an identity ID_i. If the index of identity ID_i satisfies $i \neq j^*$, then the challenger B picks $t_{i0}, t_{i1} \in Z_q$ randomly to set up $H_0(\text{ID}_i\|0) = t_{i0}P = h_{i0}$ and $H_0(\text{ID}_i\|1) = t_{i1}P = h_{i1}$. Otherwise, the challenger B uniformly samples $t*1$ and $t*0$ from Z_q to set up

$$\begin{aligned} h_{i0} &= H_0\left(\text{ID}_i \| 0\right) = t_0^* bP, \\ h_{i1} &= H_0\left(\text{ID}_i \| 1\right) = -t_1^* bP. \end{aligned} \tag{12}$$

In the end, the 5-tuple $(\text{ID}_i, h_{i0}, h_{i1}, t_{i0}, t_{i1})$ is added in the H_0-list being initially empty.

Key Extraction Oracle. For a key extraction query, {Adv} submits an identity information ID_i to key extraction oracle. To response it, the challenger B calculates the following:

(1) If the identity index of ID_i satisfies $i \neq j^*$, then B looks for 5-tuple $(h_{i0}, h_{i1}, \mathrm{ID}_i, t_{i0}, t_{i1})$ in the H_0-list. If it exists, B sends $D_{i0} = t_{i0}aP, D_{i1} = t_{i1}aP$ to the adversary Adv; otherwise, it implicitly queries a H_1-Oracle with identity ID_i.

(2) Otherwise, B terminates it.

TagGen Oracle. If the adversary {Adv} submits 3-tuple (M, IDi, Name) to TagGen Oracle for authentication tag query, where $M = m_1 \| \ldots \| m_n$ and Name is the file identifier of data file M. To response it, the challenge B calculates the following:

(1) First of all, it searches the H_0-list to check if IDi exists. If it is, then the corresponding 5-tuple $(h_{i0}, h_{i1}, \mathrm{ID}_i, t_{i0}, t_{i1})$ in the H_0-list is returned. Otherwise, B needs to query H_0-Oracle with identity information IDi.

(2) If the identity index satisfies $i = j^*$, then the challenge B aborts it. Otherwise, it produces authentication tags on data file M by the following process:

(a) Firstly, for file identifier "Name," it picks $r_{\mathrm{name}} \in Z_q$ randomly to calculate $R_{\mathrm{name}} = r_{\mathrm{name}}P$.

(b) Next for $l = 1$ to n, it calculates

$$h_{il} = H_2\left(\mathrm{Index}_l \| \mathrm{ID}_i \| R_{\mathrm{name}} \} \| 0\right). \tag{13}$$

And then for $l = 1$ to n, B calculates data block m_1's authentication tag as

$$\delta_{il} = r_{\mathrm{name}} \cdot H_1\left(\mathrm{Index}_i \| \mathrm{Name}\right) + \left(h_{il}t_{i0} + m_l t_{i1}\right)P_{\mathrm{pub}}, \tag{14}$$

and adds $(\mathrm{ID}_i, R_{\mathrm{name}}, \{m_l, \delta_{il}\}\{1 \leq l \leq n\})$ to the Tag list which is initially empty.

(3) Finally, it returns $(\mathrm{ID}_i, R_{\mathrm{name}}, \{m_l, \delta_{il}\}\{1 \leq l \leq n\})$ to the adversary {Adv}.

Output. In the end, for a challenge information $\{(i, v_i)\}, i \in I$, the adversary Adv outputs a fake proving information (δ^*, μ^*, R^*) on data user's the corrupted file M^* in a nonneglected probability ε, where the data user's identity is ID^*. Adv wins this security game if and only if the following constraint condition holds:

(1) $\mathrm{ID}^* = \mathrm{ID}_j$

(2) $\mathrm{Prf}' = (\delta^*, R^*)$ can pass verification equation (11)

(3) $\mathrm{Prf}' \neq \mathrm{Prf}$, where $\mathrm{Prf} = (\delta, R, \mu)$ should be a legitimate proving information for the challenge information $(i, v_i), i \in I$ and the data file M which satisfies $M' \neq M$

When the adversary Adv wins this game, then we are capable of obtaining the following:

$$\begin{aligned} e(\delta^*, P) &= e(\overline{H}, R^*)e\left(\widehat{h}^* \cdot Q_{\mathrm{ID}j0} + \mu^* Q_{\mathrm{ID}j1}, P_{\mathrm{pub}}\right), \\ e(\delta, P) &= e(\overline{H}, R)e\left(\widehat{h} \cdot Q_{\mathrm{ID}j0} + \mu Q_{\mathrm{ID}j1}, P_{\mathrm{pub}}\right), \end{aligned} \tag{15}$$

where $\widehat{h} = \sum_{i \in I} v_i H_2(\mathrm{Index}_i \| R^* \| \mathrm{ID}_j \| 0)$ and $h^* = \sum_{i \in I} v_i H_2(\mathrm{Index}_i \| R^* \| \mathrm{ID}_j \| 0)$ because $\widehat{H} = \sum_{i \in I} v_i \cdot H_1(\mathrm{Index}_i \| \mathrm{Name})$ is computed by the verifier, and for the same data file, $R = R^*$ and $h^* = h$. Thus, we have

$$\begin{gathered} e(\delta - \delta^*) = e\left(P_{\mathrm{pub}}, (\mu - \mu^*)Q_{\mathrm{ID}j}\right), \\ \Downarrow \\ e(\delta - \delta^*) = e(aP, (\mu - \mu^*)(-t_1^*)bP), \\ \Downarrow \\ abP = \frac{1}{(\mu - \mu^*) \cdot t_1^*}(\delta - \delta^*). \end{gathered} \tag{16}$$

It indicates that the CDH assumption is able to be broken with nonneglected probability ε'. Apparently, it is impossible since it is a hard problem to solve the CDH problem.

Theorem 2. *For a malicious cloud server, its replay attack in our proposed auditing proposal can efficiently be resisted.*

Proof. The proof in detail is very alike with the security proof in [16]. Hence, it is left out due to the limited space.

5. Performance Evaluation

To efficiently evaluate our proposal's performance, in the following part, we show that our proposal is efficient by comparing with Yu et al.'s proposal [15] and Zhang and Dong's proposal [16] in the light of computational cost and communication overhead, where Zhang and Dong's proposal [16] which is the state-of-the-art identity-based public auditing schemes in the aspect of communication overhead.

5.1. Computation Costs. To evaluate the computation costs of our proposal, we would like to contrast our proposal with Zhang and Dong's proposal [16] and Yu et al.'s proposal [15] since the two schemes are recent two efficient ID-based public auditing schemes. We emulate the operators adopted in the three schemes on an HP-laptop computer with an Intel-Core i3-6500 CPU at 2.4 GHz processor and 8 GB RAM and all algorithms are implemented using the MIRACL cryptography library [21, 22], which is used for the "MIRACL-Authentication Server-Project Wiki" by Certivox. We employ a Super-singular elliptic curve over field GFp, which has the 160-bit modulus p and a 2-embedding degree. Moreover, in our experiments, the whole statistical results are from the mean values of 10 simulation trials.

For explicit demonstration, we use Mul_{G1} to denote point multiplication operation, and let *Hash* and *Pairing* be one hash-to-point operation from Z_q to G_1 and a bilinear pairing operation, respectively. For a public auditing protocol, the computational cost in the TagGen phase is mainly determined by computation of producing block authentication tags. To outsource a data file, the data user requires $(3n+1)$ $\text{Mul}_{G1} + n$ hash to produce block authentication tag in our construction; in Yu et al.'s proposal and Zhang et al.' s proposal, each data user needs $(n+1)\text{Hash} + (2n+2)$ Mul_{G1} and $4n$ Mul_{G1} to calculate metablock authentication tag, respectively, where n is the numbers of data block. In Figure 3, we show the simulation result of generating the block authentication tag for the size of diverse data blocks with the identical data file.

From Figure 3, we can know that, in the TagGen phase, Zhang et al.'s proposal is the most time-consuming and Yu et al.'s proposal is the most efficient. Our proposal is slightly slower than Yu et al.'s one since the algorithm to produce the block authentication tag is a public key certificate-based signature algorithm in Yu et al.'s proposal; however, the algorithm, which is used in our proposal, is the ID-based signature algorithm. Because block authentication tags for the data file can be produced in the off-line phase, it has a little influence on the whole protocol.

In auditing the verification phase, the computational cost mainly comes from verifying proof information. It is determined by the numbers of the challenge data blocks. For our construction, to check the validity of proving information, the auditor requires $3pairing + c$ Hash $+ (c+2)$ Mul_{G_1}; however, in the other two proposals, the TPA needs to execute $3Pairing + 2\text{Mul}_{G1}$ and $5Pairing + (c+2)$ $\text{Mul}_{G1} + c$ *Hash* to check the integrality of the stored file in cloud, respectively, where c expresses the size of the challenge subset. In Table 2, we give their comparison of computational time in the different challenge subset.

According to Table 2, we infer that the proposal in [16] is the most efficient. Our proposal is slightly more efficient than the proposal in [15]. However, the proposal in [16] is shown to be insecure, and its detail attack is shown in Appendix. At the same time, we also find that the TPA's computational costs grow linearly with the size of the challenge subset.

5.2. Communication Cost. In a data audit system, communication costs mainly come from two aspects. On the one hand, it is from the outsource phase of the datafile; on the other hand, it is from the auditing phase. In the outsource phase, data owner uploads data file and the corresponding meta-authentication tags. As far as our proposal, the data owner wants to upload $(n+1)|G\_1| + |M|$ bits to cloud storage server; however, in the proposals [15, 16], the data owner wants to upload $(n+3)|G\_1| + |M|$ bits and $2n \cdot |G\_1| + |M|$ bits, respectively. Here, $|G\_1|$ represents the bit length of an element of group G_1, $|M|$ denotes the bit length of data file, and n is the number of data blocks.

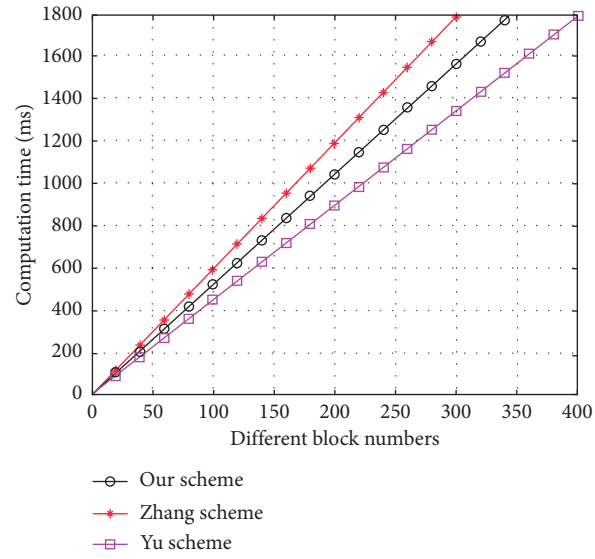

Figure 3: Computational cost of authentication tag generation for different block numbers.

Table 2: Comparison of computation time in the auditing phase.

Scheme	The number of the challenged blocks		
	$c = 300$	$c = 460$	$c = 1000$
Computation time in Yu et al.'s scheme (s)	0.644	0.9733	2.31
Computation time in Zhang et al.'s scheme (s)	0.109	0.161	0.333
Computation time in our scheme (s)	0.616	0.9367	2.02

In the auditing phase, communication costs are mainly from the challenge information and proving information transmitting between the TPA and cloud storage servers. In our scheme, the challenge information Chall is $|Z\_q| + |I|$ bits, proving information is $3 \cdot |G\_1|$ bits, and thus, the total communication overhead is $|Z\_q| + |I| + 3 \cdot |G\_1|$ bits, where $|Z\_q|$ represents the bit length of an element in group Z_q and $|I|$ is the size of the challenge subset. In the proposal [16], the total communication cost is $2|Z\_q| + |I| + 2 \cdot |G\_1|$ bits in the auditing phase; in the proposal [15], the total communication costs is $(2 + |I|)|Z\_q| + |I| + 5 \cdot |G\_1|$ bits in the auditing phase. Their comparison in detail is shown in Table 3.

As shown in Table 3, our scheme has the least communication overhead among three schemes.

5.3. Security Comparison. According to *Theorem 1*, we know that our scheme is provably secure against the vicious cloud server in the computational Diffie–Hellman assumption, and it has tight security reduction. For Yu et al.'s proposal in [15], their proposal is also provably secure against the vicious cloud server under the CDH assumption. However, for Zhang and Dong's

Table 3: Comparison of communication overhead and security among three schemes.

Scheme	Challenge information (bits)	Proof information (bits)	Total (bits)	Security
Zhang et al.'s scheme	$\|I\| + \|Z_q\|$	$\|Z_q\| + 2\|G_1\|$	$2\|Z_q\| + 2\|G_1\| + \|I\|$	No
Our scheme	$\|I\| + \|Z_q\|$	$3\|G_1\|$	$\|Z_q\| + 3\|G_1\| + \|I\|$	Yes
Yu et al.'s scheme	$\|I\|\|Z_q\| + \|I\| + 2\|G_1\|$	$2\|Z_q\| + \|I\| + 3\|G_1\|$	$(2 + \|I\|)\|Z_q\| + 3\|G_1\| + \|I\|$	Yes

proposal [16], it is shown to be insecure against the vicious cloud server attack. A vicious cloud server is capable of deleting the whole file without being conscious of the TPA, and the detail security analysis is given in Appendix.

6. Conclusion

In this work, we present a novel identity-based public audit system by merging the homomorphic authentication technique in the ID-based cryptography into the audit system. Our proposal overcomes the security problem and efficiency problems which have existed in the ID-based public audit systems. Finally, the proposal is proven to be secure, their security is tightly relevant to the classical CDH security assumption. By compared with two efficient ID-based schemes, our scheme outperforms those two ID-based schemes under the condition of overall considering computation complexity, communication overhead, and security.

Appendix

For a vicious s cloud storage server, its attack is conducted as follows:

(1) Suppose that FM is an outsourced file. It is firstly partitioned into n blocks, i.e., $\mathrm{FM} = m_1 \| \ldots \| m_n$. And $\pi_i = (\delta_i, R_i)$ is a meta-authentication signature of each block m_j, for $j = 1, \ldots, n$.

(2) For a vicious cloud storage cloud, it calculates $h_i = H_2(m_i \| \mathrm{ID} \| R_i \| 0)$, $1 \le i \le n$.

(3) Subsequently, it uniformly samples a number $r_a \in Z_q$ to compute $\widehat{R}_i = m_i R_i + r_a m_i P$ and $\widehat{\delta}_i = \delta_i + r_a m_i T$ for $i = 1, 2, \ldots, n$. And delete all data blocks m_i from the cloud server for $i = 1, \ldots, n$.

(4) After the challenge information $\mathrm{Chall} = (\rho, I)$ is received; the vicious remote server in cloud firstly calculates $v_i = \rho^i \bmod q$ for $i \in I$.

(5) Then, it calculates $\delta^* = \sum_{i \in I} v_i \widehat{\delta}_i$, $h^* = \sum_{i \in I} v_i h_i$, and $R^* = \sum_{i \in I} v_i \cdot \widehat{R}_i$. Note that h_i has been calculated in step 2.

(6) The forged proof information is $\mathrm{Prf}^* = (\delta^*, h^*, R^*)$. Since the forged proof information satisfies the relation

$$
\begin{aligned}
e(\delta^*, P) &= e\left(\sum_{i \in I} (v_i \delta_i + v_i r_a T), P\right) \\
&= e\left(\sum_{i \in I} v_i (r_i m_i T + h_i D_{\mathrm{ID0}} + h'_i D_{\mathrm{ID1}}) \right. \\
&\quad \left. + v_i r_a m_i T, P\right) \\
&= e\left(\sum_{i \in I} v_i (r_i m_i T + r_a m_i T), P\right) \\
&\quad \cdot e\left(\sum_{i \in I} v_i (h_i D_{\mathrm{ID0}} + h'_i D_{\mathrm{ID1}}), P\right) \\
&= e\left(T, \sum_{i \in I} v_i (r_i + r_a) m_i P\right) \\
&\quad \cdot e\left(\left(\sum_{i \in I} (v_i h_i) D_{\mathrm{ID0}} + \sum_{i \in I} (v_i h'_i) D_{\mathrm{ID1}}\right), P\right) \\
&= e(T, R^*) \cdot e\left(\left(h^* D_{\mathrm{ID0}} + h'^* D_{\mathrm{ID1}}\right), P\right).
\end{aligned}
\tag{A.1}
$$

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We would like to thank the anonymous referees for their invaluable suggestions in TrustCom2016. This research was supported by the Beijing Municipal Natural Science Foundation (no. 4162020), Guangxi Key Laboratory of Cryptography and Information Security (no. GCIS201710), and Research Fund of Guangxi KeyLab of Multi-Source Information Mining & Security (no. MIMS16-01).

References

[1] E.-C. Chang and J. Xu, "Remote integrity check with dishonest storage server," in *Proceedings of European Symposium on Research in Computer Security (ESORICS'08)*, pp. 223–237, Málaga, Spain, October 2008.

[2] A. Juels and B. S. Kaliski Jr., "Pors: proofs of retrievability for large files," in *Proceedings of 14th ACM Conference on*

Computer and Communications Security (CCS'07), pp. 584–597, Alexandria, VA, USA, October-November 2007.

[3] S. G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrustedstores," in *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS'07)*, pp. 598–609, Alexandria, VA, USA, October-November 2007.

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

[5] J. Zhang, W. Tang, and J. Mao, "Efficient public verification proof of retrievability scheme in cloud," *Cluster Computing*, vol. 17, no. 4, pp. 1401–1411, 2014.

[6] H. Wang, "Identity-based distributed provable data possession in multicloud storage," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 328–340, 2015.

[7] J. Zhang and H. Meng, "Comment on id-based remote data integrity checking with data privacy preserving," *IOP Conference Series: Materials Science and Engineering*, vol. 231, article 012006, 2017.

[8] C. Liu, R. Ranjan, C. Yang, X. Zhang, L. Wang, and J. Chen, "MuR-DPA: top-down levelled multi-replica Merkle Hash tree based secure public auditing for dynamic big data storage on cloud," *IEEE Transactions on Computers*, vol. 64, no. 9, pp. 2609–2622, 2015.

[9] N. Kaaniche, A. Boudguiga, and M. Laurent, "ID-based cryptography for secure cloud data storage," in *Proceedings of IEEE Sixth International Conference on Cloud Computing*, pp. 375–382, Santa Clara, CA, USA, 2013.

[10] F. Sebe, J. Domingo-Ferrer, A. Martnez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 8, pp. 1034–1038, 2008.

[11] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proceedings of International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'08)*, pp. 90–107, Melbourne, VIC, Australia, December 2008.

[12] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Computing*, vol. 16, no. 1, pp. 69–73, 2012.

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

[14] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditingto keep online storage services honest," in *Proceedings of 11th USENIX Workshop on Hot Topics in Operating Systems (HOTOS'07),* G. C. Hunt, Ed., San Diego, CA, USA, May 2007.

[15] Y. Yu, Y. Zhang, Y. Mu, W. Susilo, and H. Liu, "Provably secure identity based provable data possession," in *Proceedings of International Conference on Provable Security (ProvSec 2015)*, pp. 310–325, Kanazawa, Japan, November 2015.

[16] 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.

[17] B. Wang, B. Li, and H. Li, "Public auditing for shared data with efficient user revocation in the cloud," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM 2013)*, pp. 2904–2912, Turin, Italy, April 2013.

[18] Q. Wang, C. Wang, K. Ren, W. 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.

[19] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM 2010)*, pp. 525–533, San Diego, CA, USA, March 2010.

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

[21] https://libraries.docs.miracl.com/miracl-explained/what-is-miracl.

[22] C. Gritti, W. Susilo, and T. Plantard, "Efficient file sharing in electronic health records," in *Proceedings of International Conference on Information Security Practice and Experience (ISPEC 2015), LNCS*, vol. 9065, pp. 499–513, Beijing, China, May 2015.