Journal of Computers Vol. 31 No. 3, 2020, pp. 154-167
doi:10.3966/199115992020063103012

# Pseudo-ID-based Public Auditing with Privacy-Preserving for Cloud Storage

Xue-Yan Liu[1*], Xiao-Mei He[1], Ting-Ting Lu[1]

[1] College of Mathematics and Statistics, Northwest Normal University, Lanzhou, China
liuxy@nwnu.edu.cn, 18909318819@163.com, 37599487@qq.com

Received 16 November 2018; Revised 18 March 2019; Accepted 26 April 2019

Abstract. Public verification of data integrity is crucial in promoting the serviceability of cloud storage systems. Auditing technique is an efficient tool to check the integrity of the remotely stored data. However, the general audit mechanism only considers the confidentiality of outsourced data, and ignores the anonymity of data owners. The exposure of data ownership causes outsourced data become vulnerable to attacks. To address this problem the study presents a new model of pseudo-ID-based auditing protocol. We propose the first pseudo-ID-based public auditing scheme (PIDPA) for cloud data integrity, which is provably secure based on the computational Diffie-Hellman (CDH) problem. In particular, the introduction of pseudo-identity ensures the anonymity of data owners and allows a key generation center (KGC) to track the real identity of data owners when disputes arise. Moreover, full batch auditing is supported in the multi-user scenario on the basis of homomorphic signature. Theoretical analyses prove that the proposed PIDPA scheme is secure while maintaining the desirable security properties, and experimental results show that our scheme is efficient to audit data integrity in the cloud.

Keywords: batch auditing, cloud storage, data integrity, provable security, pseudo identity

## 1 Introduction

With the explosive growth in volume of data worldwide, cloud storage has been widely used and has provided major benefits and conveniences for individuals and organizations. However, cloud storage involves many security issues, such as data access [1], search over encrypted outsourced data [2], data integrity [3-4], and privacy-preservation [5-6]. Ensuring data integrity in the cloud is a key issue in cloud storage [7]. As a data user deletes local data and loses control of his/her data after uploading, the user may be concerned that his/her data would be lost or corrupted due to hardware errors and software bugs. Furthermore, the cloud storage server may deliberately tamper or delete the data for illegal purposes. Therefore, the integrity of outsourced data in the cloud should be checked.

To address this security issue, various auditing schemes have been proposed in recent years. Ateniese et al. [8] presented a public auditing scheme to check the integrity of remote data, which is the base of integrity checking schemes, however, this scheme cannot guarantee that the data can be retrieved. Shacham and Waters [9] established the compact proofs of retrievability concept based on BLS signature [10]. Then, considerable schemes have been proposed to check the integrity of outsourced data without downloading the entire files [4, 11-21]. Most of existing public auditing schemes are based on the public key infrastructure (PKI) [8, 13]. In this infrastructure, a certificate generated by a third party is required to bind a user's identity and the associated public key. The user required to manage his/her public key certificate. Thus, certificate management results in high computation cost, especially in a multi-user setting, which makes the previous schemes based on PKI unsuitable.

The concept of identity-based public key cryptography (ID-based PKC) [22] was presented to solve certificate management problem. In the ID-based PKC, the identity of a user is his/her public key, and this private key is generated by KGC based on the public key. Thus, no certificate management is

[*] Corresponding Author

required. Zhao et al. [16] proposed an identity-based public auditing concept based on bilinear pairing. In their scheme, the user's public key is his/her identity, and the private key is generated by the KGC based on the user's identity. Subsequently, Wang et al. [20] established a real ID-based auditing scheme that was proved to be safe in their security model. Tan et al. [23] presented an ID-based scheme: called NaEPASC, which simplifies key management and alleviates the user burden. However, Wu et al. [24] indicated that NaEPASC is vulnerable to signature forgery attacks in the setup phase. Zhang et al. [4] proposed an ID-based public auditing scheme that provids batch auditing. However, He et al. [25] showed two concrete attacks that break the data integrity of the scheme. These ID-based public auditing schemes [16, 20, 23] experience the drawback of key escrow problem.

Wang et al. [17] proposed a certificateless public auditing scheme to overcome certification management and key escrow problems. The private key of a user in the certificateless public cryptography consists of two parts, namely, a secret key generated by the user and a partial private key generated by a KGC. Thus, the aforementioned certificate management and key escrow problems have been solved. Generally, the user is limited by the available computational resources. Thus, an authorized third party (TPA) is introduced to audit the data integrity rather than the data owner. The TPA should not obtain any knowledge of the audited data during auditing because he/she is a third party. In addition, the TPA and other users should not obtain the identity of the data owner; otherwise, the frequently audited data may experience many attacks and attract numerous attackers. In other words, exposure of data owner identity makes the outsourced data vulnerable to attacks. For example, the data content can be inferred by guessing and collecting information based on the identity of the data user. Unfortunately, anonymity of the data user (data owner) is not considered in the certificateless public cryptography.

However, absolute anonymity may cause other security issues for the users and data owners. For instance, a malicious data owner can damage part of shared data for some illegal reasons, and the absolute anonymity of identity can allow the data owner to escape responsibility. Thus, the absolute anonymity is unsuitable. Obviously, traceability is equally important to a public auditing scheme for shared data.

On this basis, the anonymity and traceability of data owner identity in the public auditing scheme are crucial. To overcome these problems, we propose the first pseudo-ID-based public auditing protocol (PIDPA) by hiding the real identity of data users. PIDPA is proven to be secure on the hardness of computational Diffie-Hellman (CDH) problem. Our major contributions are summarized as follows:

(1) We present a pseudo-ID-based public auditing scheme (PIDPA). Pseudo identity is introduced to ensure the anonymity of data owner. A KGC can track the real identity of the data user when disputes arise. The proposed PIDPA scheme avoids certificate management and key escrow problems.

(2) In our proposed PIDPA scheme, a real batch auditing is achieved in a multi-user scenario based on homomorphic signature.

(3) We perform a security analysis of our PIDPA scheme to demonstrate its provable secure. We also conduct performance evaluation and comparison with a previous scheme.

The rest of this paper is organized as follows. Some preliminaries are presented in Section 2. Section 3 describes system model and security requirements. We describe the proposed PIDPA scheme in Section 4. We present a security analysis and a performance analysis in Section 5 and Section 6, respectively. Finally, we make a conclusion.

## 2 Preliminaries

Let $G$ and $G_T$ be an additive group and a multiplicative group with the same prime order $q$. A map $e: G \times G \to G_T$ is called a bilinear pairing if all of the following three conditions hold.

(1) Bilinear: For any two random points $P, Q \in G$ and two random elements $a, b \in Z_q$, the equation $e(aP, bQ) = e(P, Q)^{ab}$ could be got.

(2) Nondegeneracy: There is a point $P \in G$ such that $e(P, P) \neq 1$.

(3) Computability: For any two random points $P, Q \in G$, $e(P, Q)$ could be calculated efficiently in polynomial time.

Discrete Logarithm Problem: For two random points $P, Q \in G$, it is difficult to calculate integer

$x \in Z_q$ to get equation $Q = xP$.

Computational Diffie-Hellman (CDH) problem: For two random points $R, Q \in G$, we cannot calculate point $abP$ in polynomial time, where $R = aP$, $Q = bP$, and $a, b \in Z_q$ are two unknown elements.

## 3 System Model and Security Requirements

### 3.1 System Model

There are four entities in our proposed PIDPA scheme, i.e., the third public auditor (TPA), Key generate center (KGC), data user and the cloud storage server (CSS). Their roles and interactions among them are showed in Fig. 1.

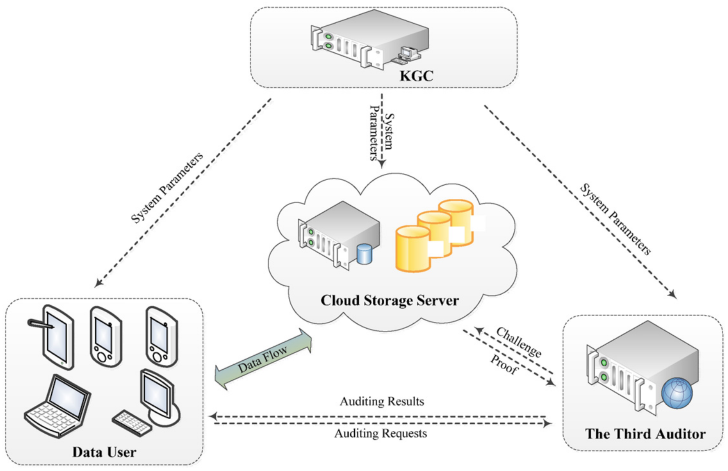

**Fig. 1.** The cloud storage model

KGC: It is a trusted party to responsible to generate system parameters and send them to the other three entities, and keep user's real identity and assists generate user's private key.

TPA: It is a honest-but curious party with more computation capabilities to provide honestly data auditing service for users.

Data user: It is a legal user of the system who has a large amount of data files to be outsourced to the cloud storage foe data maintenance and computation.

CSS: It is an semi-trusted entity which has unlimited storage space and computation capability to storing and maintaining the outsourced data, and provides the data access to the data user.

### 3.2 Security Requirements

According to the existing research results [17, 20, 23], the PIDPA scheme needs to satisfy the following fundamental functions and security requirements:

- Public Verifiability: The TPA should check the integrity of outsourced data without downloading the all data or causing additional burden on the user.
- Confidentiality: The TPA and the CSS cannot recover the data of the user's data stored in the cloud throughout the whole process.
- Storage Correctness: To ensure that a TPA can audit the storage correctness by checking the integrity of the outsourced data.
- Anonymity: In order to protect the user's privacy, his real identity should be hidden from everyone except for KGC.
- Batch Auditing: To improve the computational performance when receiving multiple requests from the same data user (the same identity), or different data users (different identity, i.e. multi-user scenario) for checking the data integrity, the third auditor could execute multiple auditing delegations simultaneously.

- Traceability: When disputes arise, the KGC can track the real identity of data user.

In addition to the above fundamental security requirements, the proposed new scheme should resist the following attacks.

- Delete/Modification Attacks: The malicious CSS may discard a challenged data block or its meta-data and replace them with another pair of data-block and meta-data in order to pass the auditing.
- Forge Attacks: The malicious CSS may forge the block meta-data resulting in improper and unsatisfactory data auditing.
- Replay Attacks: The malicious CSS generates the proof using non-updated or previous data, previous proof or other information, without querying data owner's actual data.

### 3.3 Security Model

In our PIDPA scheme, the KGC is a trusted authority, so we mainly consider that the semi-trusted CSS may launch the attack. The security of the auditing scheme is defined by a data integrity checking game played between an adversary $A$ who plays the role of the CSS and a challenger $C$ to solve CDH problem. The game has the following phases:

- **Setup:** when $C$ executes **Setup** algorithm, it gets $param$ and $Msk$. Then, $C$ returns $param$ to $A$, and keeps $Msk$ secret.
- **Queries:** $A$ makes as many queries to the challenger $C$, including **Key extraction** queries and **TagGen** queries.
  - **Key extraction queries:** Upon receiving a query with identity $PId_i$, $C$ returns $\{x_i, d_i\}$ to $A$.
  - **TagGen queries:** Upon receiving a query with a user's identity $PId_i$ and data $C_i$ with identity $id_i$, $C$ executes TagGen algorithm and returns $\{C_i, id_i, R_i\}$ to $A$.
- **Output:** Finally, $A$ outputs an authentication tag $(C^*, R^*)$ of a subset $I^* = \{i_1^*, i_2^*, \cdots, i_l^*\}$ as its forgery. The adversary $A$ wins the game if it satisfies the following restriction conditions:

  (1) True $\leftarrow$ Verifying $(PId_u^*, C^*, R^*)$;

  (2) If $PId_u^*$ has never been queried during the Key extraction queries;

  (3) At least one element $i_j^* \in I$ has never been sent to the **TagGen** queries.

## 4 Proposed PIDPA Scheme

In this section, we will put forward a public auditing scheme without key escrow, and pseudo identity is introduced. Our PIDPA scheme consists six algorithms: **Setup**, **Key Extraction**, **TagGen**, **Challenge**, **ProofGen** and **Verifying**. Fig. 2 exhibits the main auditing process.

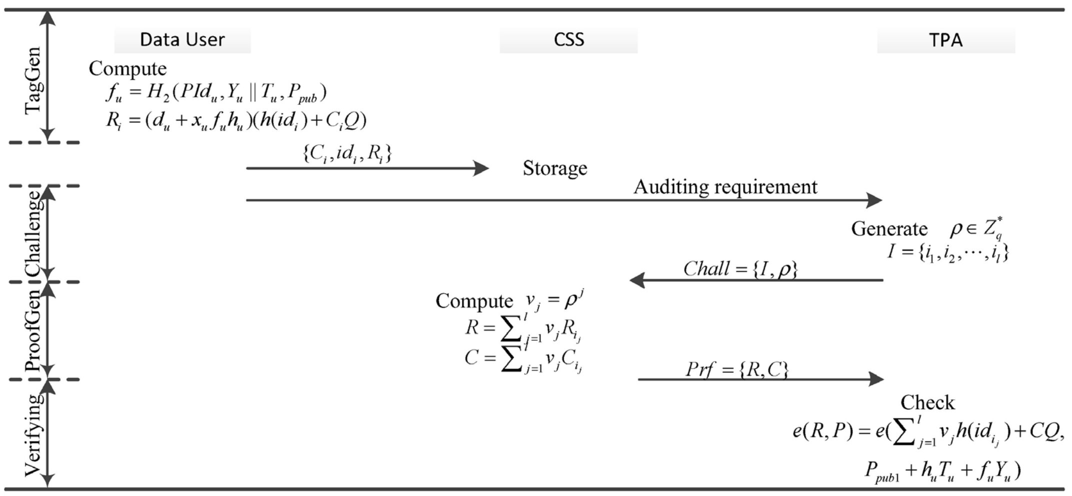
Data User
CSS
TPA
TagGen
Challenge
ProofGen
Verifying
Compute
$f_u = H_2(PId_u, Y_u \| T_u, P_{pub})$
$R_i = (d_u + x_u f_u h_u)(h(id_i) + C_i Q)$
$\{C_i, id_i, R_i\}$
Storage
Auditing requirement
Generate $\rho \in Z_q^*$
$I = \{i_1, i_2, \cdots, i_l\}$
$Chall = \{I, \rho\}$
Compute $v_j = \rho^j$
$R = \sum_{j=1}^{l} v_j R_{i_j}$
$C = \sum_{j=1}^{l} v_j C_{i_j}$
$Prf = \{R, C\}$
Check
$e(R, P) = e(\sum_{j=1}^{l} v_j h(id_{i_j}) + CQ, P_{pub1} + h_u T_u + f_u Y_u)$

**Fig. 2.** Auditing process of our proposed PIDPA scheme

The detailed algorithms are followed:

**Setup:** For KGC, it sets up the following system parameters. Given a security parameter $k$, it selects two groups $G$ and $G_T$ of the same prime order $q$, $P$ is the generator of $G$, $h:\{0,1\}^* \to G$, $H:\{0,1\}^* \to Z_q^*$, $H_0: G\times G\times G \to Z_q^*$, $H_1: G_T \to Z_q^*$, $H_2: G_T \times G \times G \to Z_q^*$ are hash functions. And randomly selects $s \in Z_q^*$, and compute $P_{pub}=\{P_{pub1}, P_{pub2}\}$, where $P_{pub1}=s^{-1}P$, $P_{pub2}=sP$, and $Q=h(P_{pub1})$. Finally, KGC publishes the following system parameters

$$Param=\{G,G_T,q,e,P,h,H,H_0,H_1,H_2,P_{pub},Q\},$$

and secretly keeps his master secret key

$$Msk=\{s\}.$$

**Key extraction:** A data user U whose real identity is $ID_U$, random chooses $x_U \in Z_q^*$ as his private key, and computes $h_{ru}=h(ID_U)$, $Y_{ru}=x_u h_{ru}$. U chooses $v_u \in Z_q^*$ randomly, computes $T_{ru}=v_u h_{ru}$, $c=H_o(T_{ru},Y_{ru},P_{pub2})$, $\theta=v_u+cx_u$, and send $\{ID_u,Y_{ru},T_{ru},\theta\}$ to KGC.

KGC computes $c=H_o(T_{ru},Y_{ru},P_{pub2})$ and $h_{ru}=h(ID_U)$ after receiving $\{ID_u,Y_{ru},T_{ru},\theta\}$, and checks $\theta h_{ru}=(v_u+cx_u)h_{ru}=v_u h_{ru}+cx_u h_{ru}=T_{ru}+cY_{ru}$, which is aborted when verification fails. Otherwise, KGC randomly chooses $t_u \in Z_q^*$, and computes $T_u=t_u^{-1}P_{pub2}=st_u^{-1}P$, $PId_u=e(Y_{ru},T_u)=e(h_{ru},P)^{x_u st_u^{-1}}$, which is the pseudo identity of user U. Then KGC computes $d_u=s^{-1}+s\cdot t_u^{-1}\cdot h_u$, where $h_u=H_1(PId_u)$. KGC sends $d_u$ and $T_u$ to U.

U computes $PId_u=e(Y_{ru},T_u)=e(h_{ru},P)^{x_u st_u^{-1}}$ as his public identity, $h_u=H_1(PId_u)$, $Y_u=x_u h_u P$ and $T_u$ are his public keys, keeps his private keys $\{x_u,d_u\}$ secretly.

**TagGen:** For data $M$, encrypted to be $C=E_k(M)$ under a certain encryption algorithm and divided into $n$ blocks $\{C_1,C_2,\cdots,C_n\}$. Data user U with pseudo identity $PId_u$, private key $\{x_u,d_u\}$ runs this algorithm to generate a tag for each $C_i$, $i\in\Omega$, $\Omega=\{1,2,\cdots,n\}$.

U computes $f_u=H_2(PId_u,Y_u \| T_u,P_{pub})$, $R_i=(d_u+x_u f_u h_u)(h(id_i)+C_iQ)$. Finally, he sends $\{C_i,id_i,R_i\}$ to the CSS.

Upon receiving the outsourced data, the CSS first aggregates the receiving data: $R=\sum_{i=1}^n R_i$, $C=\sum_{i=1}^n C_i$ and then ensures the correctness of the received data by checking the equation $e(R_i,P)\overset{?}{=}e(\sum_{i=1}^n h(id_i)+CQ,P_{pub1}+h_uT_u+f_uY_u)$. If the equation holds, the CSS storages the received data.

$$\begin{aligned}e(R,P)&=e(\textstyle\sum_{i=1}^n(d_u+x_uf_uh_u)(h(id_i)+C_iQ),P)\\&=e((d_u+x_uf_uh_u)\textstyle\sum_{i=1}^n(h(id_i)+C_iQ),P)\\&=e(\textstyle\sum_{i=1}^n h(id_i)+\sum_{i=1}^n C_iQ,(d_u+x_uf_uh_u)P)\\&=e(\textstyle\sum_{i=1}^n h(id_i)+\sum_{i=1}^n C_iQ,(s^{-1}+s\cdot t_u^{-1}\cdot h_u+x_uf_uh_u)P)\\&=e(\textstyle\sum_{i=1}^n h(id_i)+CQ,P_{pub1}+h_uT_u+f_uY_u).\end{aligned}$$

**Challenge phase:** The TPA generates a random number $\rho\in Z_q^*$ and a random subset $I=\{i_1,i_2,\cdots,i_l\}$, $i_j\in\Omega, j=1,2,\cdots,l.$ Then the auditing challenge is:

$$Chall=\{I,\rho\}.$$

**ProofGen:** After receiving $Chall=\{I,\rho\}$, CSS compute $v_j=\rho^j$, $R=\sum_{j=1}^{l}v_jR_{i_j}$, $C=\sum_{j=1}^{l}v_jC_{i_j}$, CSS sends $Prf=\{R,C\}$ to TPA.

**Verifying:** TPA computes $f_u=H_2(PId_u,Y_u,P_{pub2})$ and $v_i=\rho^i$, then verifies the equation $e(R,P)=e(\sum_{j=1}^{l}v_jh(id_{i_j})+CQ,P_{pub1}+h_uT_u+f_uY_u)$. If the verification fails, then the data integrity is corrupted.

**Correctness:** For a CSS, if it honestly responds to the challenge with $Prf$, then it must be accepted. Since

$$\begin{aligned}e(R,P)&=e(\sum_{j=1}^{l}v_j(d_u+x_uf_uh_u)(h(id_{i_j})+C_{i_j}Q),P)\\&=e((d_u+x_uf_uh_u)\sum_{j=1}^{l}v_j(h(id_{i_j})+C_{i_j}Q),P)\\&=e(\sum_{j=1}^{l}v_jh(id_{i_j})+\sum_{j=1}^{l}v_jC_{i_j}Q,(d_u+x_uf_uh_u)P)\\&=e(\sum_{j=1}^{l}v_jh(id_{i_j})+\sum_{j=1}^{l}v_jC_{i_j}Q,(s^{-1}+s\cdot t_u^{-1}\cdot h_u+x_uf_uh_u)P)\\&=e(\sum_{j=1}^{l}v_jh(id_{i_j})+CQ,P_{pub1}+h_uT_u+f_uY_u).\end{aligned}$$

**Multi-user case.** If multiple users initiate audit request, the TPA can complete the auditing task at one time. Given $m$ data users, and the auditing challenge is:

$$Chall=\{I,\rho\},$$

where $I=\bigcup_{k=1}^{m}I^k, I^k=\{i_1,i_2,\cdots,i_{l^k}\}$, $i_j\in\Omega^k, j=1,2,\cdots,l^k$.

The CSS produces corresponding proof information $\Pr f=\{R,C^1,C^2,\cdots,C^m\}$.

where $R=\sum_{k=1}^{m}R^k=\sum_{k=1}^{m}\sum_{k=1}^{l^k}v_{kj}R_{ki_j}$, $C^k=\sum_{k=1}^{l^k}v_{kj}C_{ki_j}$, $v_{ki}=(\rho)^{i_j^k}$, $k=1,2,\cdots,m$.

Upon receiving the proof information from the CSS, the TPA computes $f_u^k=H_2(PId_u^k,Y_u^k,P_{pub})$ and $v_{ki}=(\rho)^{i_j^k}$, then checks the validity of the following equation:

$$e(R,P)=\prod_{k=1}^{m}e(\sum_{j=1}^{l_k}v_{kj}h(id_{ki_j})+C^kQ^k,P_{pub1}+h_u^kT_u^k+f_u^kY_u^k). \qquad \textbf{(1)}$$

If equation (1) hold, the CSS provides correct proof. Correctness verification is followed.

$$\begin{aligned}e(R,P)&=e(\sum_{k=1}^{m}R^k,P)\\&=e(\sum_{k=1}^{m}\sum_{k=1}^{l^k}v_{kj}R_{ki_j},P)\\&=\prod_{k=1}^{m}e(\sum_{k=1}^{l^k}v_{kj}R_{ki_j},P)\\&=\prod_{k=1}^{m}e(\sum_{j=1}^{l_k}v_{kj}h(id_{ki_j})+C^kQ^k,P_{pub1}+h_u^kT_u^k+f_u^kY_u^k).\end{aligned}$$

## 5 Security Analysis

In this section, we first demonstrate that our PIDPA scheme is secure against the adversary $A$ under CDH problem. Then we prove that our scheme can resist the attacks and meet the security requirements presented in Section 3.2.

### 5.1 Security Analysis

**Theorem 1.** If there exists an adversary A which can forge a valid authentication tag with probability $\varepsilon'$, then CDH problem can be solved with probability $\varepsilon$.

$$\varepsilon \geq \frac{1}{q_{H_1}}(1-\lambda^{l^*})(1-\frac{1}{q_{H_1}})^{q_{key}}\varepsilon' . \quad \textbf{(2)}$$

**Proof:** Assume that there exist an adversary A which an forge an authentication tag without knowing private key of the data user $ID_u$ whose public identity is $PId_u$, then we can construct another algorithm $C$ which is able to solve the CDH problem. First, let us recall the CDH problem, given $(P, Q_a = aP, Q_b = bP)$, which is random instance of the CDH problem, its goal is to compute $abP$.

In the following game, we regard the hash functions $h, H_1$ and $H_2$ as random oracles and each identity $PId_u$ can only query $H_1$ once. And the adversary $A$ can adaptively make queries, Key extraction queries and TagGen queries.

- **Setup:** Let $G$ and $G_T$ be two cyclic groups of the prime order $q$. The algorithm $C$ sets $P_{pub1} = s^{-1}P = aP$ the public key of KGC. Then it returns $(G, G_T, q, P, P_{pub}, e)$ to the adversary $A$. $C$ chooses an identity $PId_u$ as a challenge identity and answers $A$'s queries as followed:
- **Queries:**

  $H_1 - queries$: $C$ maintains a list $H_1^{list}$ of group $\{PId_i, h_i, Y_i, P_i\}$, where $H_1^{list}$ is initialized to empty. Upon receiving a query with message $PId_i$, $C$ checks whether a tuple $\{PId_i, h_i, Y_i, P_i\}$ is included in $H_1^{list}$. If so, $C$ returns $h_i$ to $A$; otherwise, $C$ generates a random $h_i \in Z_q^*$ stores $\{PId_i, h_i, Y_i, P_i\}$ in $H_1^{list}$, and returns $h_i$ to $A$.

  $H_2 - queries$: $C$ maintains a list $H_2^{list}$ of group $\{PId_i, Y_i, P_{pub}, f_i\}$, where $H_2^{list}$ is initialized to empty. Upon receiving a query with message $\{PId_i, Y_i\}$, $C$ checks whether a tuple $\{PId_i, Y_i, P_{pub}, f_i\}$ is included in $H_2^{list}$. If so, $C$ returns $f_i$ to $A$; otherwise, $C$ generates a random number $f_i \in Z_q^*$, stores $\{PId_i, Y_i, P_{pub}, f_i\}$ in $H_2^{list}$, and returns $f_i$ to $A$.

  $h - queries$: $C$ maintains a list $h^{list}$ of group $\{\mathrm{i}d_i, C_i, c_i, z_i, Z_i\}$, where $h^{list}$ is initialized to empty. Upon receiving a query with $id_i$, $C$ checks whether a tuple $\{\mathrm{i}d_i, C_i, c_i, z_i, Z_i\}$ is included in $h^{list}$. If so, $C$ returns $h(id_i) = Z_i - C_i Q$ to $A$; otherwise, $C$ chooses a random bit $c_i \in \{0,1\}$ such that $\Pr[c_i = 0] = \lambda$, where $\lambda \in (0,1)$. $C$ generates a random number $c_i \in Z_q^*$. If $c_i = 0$, $C$ computes $Z_i = z_i P$; otherwise, $C$ computes $Z_i = z_i Q_b$. Finally, $C$ stores $\{\mathrm{i}d_i, C_i, c_i, z_i, Z_i\}$ in $h^{list}$ and returns $h(id_i) = Z_i - C_i Q$ to $A$.

  - **Create-User:** $C$ maintains a list $User^{list}$ of group $\{PId_i, h_i, Y_i, x_i, T_i\}$, where $User^{list}$ is initialized to empty. Upon receiving a query with $PId_i$, $C$ checks whether a tuple $\{PId_i, h_i, Y_i, x_i, P_i\}$ is included in $User^{list}$. If so, $C$ returns $\{Y_i, T_i\}$ to $A$. If $PId_i = PId_u$, generates two random numbers $t_i, x_i \in Z_q^*$, computes $Y_i = x_i h_i P$ and $T_i = t_i P_{pub2}$; otherwise, $C$ generates three random numbers $s_i, x_i, \omega_i \in Z_q^*$, computes $Y_i = x_i h_i P$ and $T_i = h_i^{-1}(s_i \omega_i P - P_{pub1})$, stores $\{PId_i, h_i, Y_i, x_i, T_i\}$ in $User^{list}$, and returns $\{Y_i, T_i\}$ to $A$.
  - **Key extraction queries:** Upon receiving a query with identity $PId_i$, $C$ checks whether $PId_i$ and $PId_u$ are equal. If $PId_i = PId_u$, $C$ reports failure and terminates. Otherwise, $C$ looks up $User^{list}$ for tuple $\{PId_i, h_i, Y_i, x_i, P_i\}$ and returns $x_i$ to $A$.
  - **TagGen queries:** Upon receiving a query with a user's identity $PId_i$ and data $C_i$ with identity $id_i$, $C$ makes a query $h$ with $id_i$ and gets tuple $\{id_i, c_i, z_i, Z_i\}$. If $c_i = 0$, $C$ computes $R_i = z_i \cdot (P_{pub1} + h_i T_i + f_i P_i)$ and returns $\{C_i, id_i, R_i\}$ to $A$; otherwise, $C$ aborts the game.
- **Output:** Eventually, $A$ outputs an authentication tag $(C^*, R^*)$ of a subset $I^* = \{i_1^*, i_2^*, \cdots, i_l^*\}$ corresponding to the data owner's identity $PId_u^*$ in a non-negligent probability $\varepsilon$. The adversary $A$

wins the game if it satisfies the following restriction conditions

1. $(C^*, PId_u^*)$ has never been queried during TagGen queries.
2. $PId_u^* = PId_u$ .
3. $(C^*, R^*)$ is a valid authentication tag on data $C^*$ .

If $PId_u^* \neq PId_u$ or $c_{i_j^*} = 0 (j = 1, 2, \cdots, l^*)$ , $C$ aborts the game; otherwise, $C$ looks up tables $User^{list}$ and $h^{list}$ for $\{PId_i, h_i, Y_i, x_i, P_i\}$ and $\{\mathrm{i}d_{i_j^*}, C_{i_j^*}, c_{i_j^*}, z_{i_j^*}, Z_{i_j^*}\}$ , where $j = 1, 2, \cdots, l^*$ , $C$ could get

$$e(R^*, P) = e(\sum\nolimits_{j=1}^{l^*} v_j h(id_{i_j^*}) + C^* Q, P_{pub1} + h_u^* T_u^* + f_u^* Y_u^*) .$$

Thus, we can obtain

$$abP = \frac{1}{h_u^* \cdot D}(R^* - t_u^{*-1} s^* f_u^* \cdot D \cdot Q_b - x_u^* f_u^* h_u^* \cdot D \cdot Q_b) \text{, where } D = \sum\nolimits_{j=1}^{l^*} v_j z_i .$$

It means that the adversary $A$ can solve the computational Diffie-Hellman problem with non-negligible probability $\varepsilon'$.

Now we analyze the probability that $C$ could solve the given CDH problem by using $A$ as his subfunction. According to the game, we should analysis the following three events related to the success of $C$ .

$E_1$ : $C$ doesn't abort in query Key extraction.

$E_2$ : $A$ output a legal proof of a subset $I^* = \{i_1^*, i_2^*, \cdots, i_{l^*}^*\}$ with at least one $c_{i_j^*} = 1, j \in \{1, 2, \cdots, l^*\}$ .

$E_3$ : After $E_2$ happens, we have $PId_u^* = PId_u$ .

The probability that $C$ could solve the given CDH problem is

$$\Pr[E_1 \wedge E_2 \wedge E_3] = \Pr[E_1]\Pr[E_2 \mid E_1]\Pr[E_3 \mid E_1 \wedge E_2] \geq \frac{1}{q_{H_1}}(1 - \lambda^{l^*})(1 - \frac{1}{q_{H_1}})^{q_{key}} \varepsilon' .$$

Where $\Pr[E_1] \geq (1 - \frac{1}{q_{H_1}})^{q_{key}}$ , $\Pr[E_2 \mid E_1] \geq (1 - \lambda^{l^*})\varepsilon'$ , $\Pr[E_3 \mid E_1 \wedge E_2] \geq \frac{1}{q_{H_1}}$ , $q_{H_1}$ and $q_{key}$ denote the number of $H_1$ and Key extraction queries, respectively.

## 5.2 Other Security Analysis

This section shows that the proposed PIDPA scheme could satisfy the security requirements of auditing schemes.

**Theorem 2.** Our auditing protocol can resist the replay attack for the cloud server.

**Proof:** For the CSS, if a challenged data block $C_j$ or its authentication tag $R_j$ is corrupted or not up-to-data on the server, the CSS may execute the replace attack to pass the audit. It might use another pair $(C_t, R_t)$ to replace the challenged pair $(C_j, R_j)$ . Then, the proof information is calculated as

$$R^* = v_j R_t + \sum_{i \in I, i \neq j} v_i R_i \text{ , } C^* = v_j C_t + \sum_{i \in I, i \neq j} v_i C_i \text{ ,}$$

According to the verification equation, we have the following relation

$$\begin{aligned} e(R^*, P) &= e(v_j R_t + \sum_{i \in I, i \neq j} v_i R_i, P) \\ &= e(v_j h(id_t) + \sum_{i \in I, i \neq j} v_i h(id_i) + C^* Q, P_{pub} + h_u T_u + f_u Y_u) \\ &= e(v_j (h(id_t) - h(id_j)) + \sum_{i \in I} v_i h(id_i) + C^* Q, P_{pub} + h_u T_u + f_u Y_u) \end{aligned} \quad \textbf{(3)}$$

If $v_j(h(id_t)-h(id_j))=0$, then equation (3) must hold. However, due to the collision resistance of hash function, $h(id_t)-h(id_j)\neq 0$. Therefore, the proof information cannot pass the auditing, and our PIDPA scheme can resist the replace attack.

**Theorem 3.** Our auditing protocol can resist the delete/modification for the cloud server.

**Proof:** For the CSS, if any outsourced data block $C_j$ is deleted or modified, then it cannot pass the auditing since the verification equation doesn't hold. The CSS may use its $C_t$ replace $C_j$, and computes

$$R^*=v_jR_t+\sum_{i\in I,i\neq j}v_iR_i\ ,\ C^*=v_jC_t+\sum_{i\in I,i\neq j}v_iC_i\ \text{, where } R_t=(d_u^{'}+x_u^{'}f_uh_u)(h(id_j)+C_tQ)\ .$$

According to the verification equation, if the proof pass verification, we have the relation

$$\begin{aligned} e(R^*,P)&=e(v_jR_t+\sum_{i\in I,i\neq j}v_iR_i,P)\\ &=e(v_j((d_u^{'}+x_u^{'}f_uh_u)\text{-}(d_u+x_uf_uh_u))(h(id_j)+C_tQ)+(d_u+x_uf_uh_u)(\sum_{i\in I}v_ih(id_i)+C^*Q),P) \end{aligned}\ . \quad \textbf{(4)}$$

If $((d_u^{'}+x_u^{'}f_uh_u)\text{-}(d_u+x_uf_uh_u))(h(id_j)+C_tQ)=0$, that is $(d_u^{'}\text{-}d_u)+(x_u^{'}\text{-}x_u)f_uh_u=0$, then equation (4) must hold. However, $d_u=s^{-1}+s\cdot t_u^{-1}\cdot h_u$, where $t_u$ and $s$ are chosen by KGC, and $x_u$ is the user's private key, it is impossible to choose the appropriate $(x_u,s,t_u)$ or $(x_u,d_u)$ for DL problem. Thus, our PIDPA scheme can resist the delete/modification attack.

**Theorem 4.** In our auditing protocol, the TPA cannot obtain any information about the outsourced data during the whole auditing procedure.

**Proof:** During the auditing procedure, the TPA only gets $Prf=\{R,C\}$, where $R=\sum_{j=1}^{l}v_jR_{i_j}$, $C=\sum_{j=1}^{l}v_jC_{i_j}$. Though the TPA can compute $v_i,i\in I$, it cannot get any $R_i,C_i$ from the processed $\Pr f=\{R,C\}$, they are linear system with $l$ unknown variables but only 1 equations respectively. Hence, the TPA cannot get any information about the data block from the whole auditing process.

In addition to resist the above attacks, security requirements, our scheme meet the security requirements presented in Section 3.2.

- Public Verifiability: Any TPA can be allowed to check the integrity of the challenge results without downloading the outsourced or causing additional burden on the user. So, our PIDPA scheme can satisfy the requirement of public auditing.
- Storage correctness: According to the specification of our PIDAPA scheme, TPA can execute the data integrity verification process to check the correctness of outsourced data upon challenge request. So, our PIDPA scheme can satisfy the requirement of storage correctness.
- Anonymity: According to the definition in our PIAPA scheme, except for KGC, no one can learn the real identity of data user. Hence, our PIDPA scheme can satisfy the requirement of anonymity.
- Batch Auditing: According to the definition in our PIDPA scheme, the TPA could execute multiple auditing delegations simultaneously, when receiving multiple requests from the same data user(the same identity), or different data users (different identity) for checking the data integrity. Hence, our PIDPA scheme can satisfy the requirement of batch auditing.
- Traceability: If there is a dispute, only trusted authority KGC can reveal the real identity of the data user.

## 6 Performance Analysis

In this section, we evaluate the performance of the proposed PIDPA scheme in terms of communication and computation costs. Then, we demonstrate the efficiency of this scheme through simulation. We also compare the performance of our PIDPA with the scheme proposed by Zhang et al. [4].

## 6.1 Computation Cost

For the sake of simplicity, we define some notations about the mainly operation. $O_H$ : map-to-point hash function operation, $O_M$ : point multiplication operation, $O_P$ : pairing operation. The execution time of operations are shown in Table 1.

**Table 1.** Execution time of operations

| Operation | $O_H$ | $O_M$ | $O_P$ |
|---|---|---|---|
| Time/ms | 12.714 | 5.127 | 10.813 |

For our PIDPA scheme, the computation costs are due to **TagGen**, **ProofGen** phase and **Verifying** phases. We provide the time cost simulation for our PIDPA scheme in different phases, and the Table 2 presents the comparison of computation costs of our scheme and that of Zhang et al. [4]. We assume that number of data blocks is $l$. In the **TagGen** phase of our scheme, the computation cost of a single data block is $2O_M + O_H$ , and the computation cost increases with the number $l$ of data blocks, which is $l(2O_M + O_H)$. Meanwhile the computation cost in the scheme of Zhang et al. is $l(4O_M + 2O_H)$. To generate an auditing proof in **ProofGen** phase of our PIDPA scheme, the computation costs are $lO_M$ compared with $2lO_M$ in Zhang et al.'s scheme**.** In the **Verifying** phase of our PIDPA scheme, the computation costs are $(l+3)O_M + 2O_P$ compared with $(l+2)O_M + 3O_P$ and in Zhang's scheme**.** The comparison shows that the PIDPA is more efficient than Zhang et al.'s scheme in terms of computation overhead in the **TagGen**, and **ProofGen** phases. Moreover, the computation overhead in the**Verifying** is better than that of Zhang et al.'s scheme.

**Table 2.** Comparison of computation cost

| Heading evel | TagGen | ProofGen | Verifying |
|---|---|---|---|
| Zhang's [4] | $l(4O_M + 2O_H)$ | $2lO_M$ | $(l+2)O_M + 3O_P$ |
| Ours | $l(2O_M + O_H)$ | $lO_M$ | $(l+3)O_M + 2O_P$ |

## 6.2 Communication Cost

In the entire procedure, the communication costs are generated by the data user, TPA, and CSS. The data user uploads the data file and corresponding verification information to CSS, the TPA sends challenge information, and the CSS separately sends the proof. Uploading of the information by the data user is a one-time operation. Thus, its cost is not the main object.

For each **Challenge**, challenge $chall = \{I, \rho\}$ costs $l \cdot |j| + |q|$ bits, where $|j|$ is the size of a block index $j$, and $|q|$ is the element length of $Z_q$. For auditing proof $\Pr f = \{R, C\}$, the communication cost is $|G| + |q|$, where $|G|$ is the element length of $G$, and $|id|$ is the length of data block $m_i$. The comparison is shown in Table 3, which indicates that the communication cost in our PIDPA scheme is less than that in Zhang et al.'s scheme.

**Table 3.** Comparison of communication cost

| Heading evel | Data user | TPA | CSS |
|---|---|---|---|
| Zhang's [4] | $l(2|G|+|q|)$ | $l \cdot |j| + |q|$ | $2|G| + |q|$ |
| Ours | $l(|G|+|q|+|id|)$ | $l \cdot |j| + |q|$ | $|G| + |q|$ |

### 6.3 Security Properties

We exhibit a high-level comparison between PIDPA and Zhang et al.'s scheme [4] in Table 4. PIDPA supports identity privacy/anonymity, public auditing, batch auditing, and traceability without key-escrow, as well as resists replay attack and delete/modification attacks. The identity of the data user can only be traced by a trusted authorized KGC. However, He et al. [25] performed two concrete attacks on Zhang et al.'s scheme, thereby breaking the data integrity of their scheme. Thus, Zhang et al.'s scheme is not secure for cloud storage.

**Table 4.** Comparison of security properties

| Heading level | Data user Anonymity | Key-escrow | Delete/modification attack | Replay attack | Public auditing | Batch auditing | traceability |
|---|---|---|---|---|---|---|---|
| Zhang's [4] | × | √ | × | √ | √ | √ | × |
| Ours | √ | × | √ | √ | √ | √ | √ |

### 6.4 Experimental Evaluation

In this section, we evaluate the thorough experimental evaluation of our scheme. Our simulation experiment is on Intel(R) Core(TM) i7-6500U CPU at 2.5GHz and 8.00GB RAM. The algorithms are implemented using the pairing-based cryptography (PBC) library version 0.4.7-vc.zip [26]. We assume $|Z_q|$=160 bit, $|G|$=160 bit.

We also compare our proposed scheme with that of Zhang et al. in terms of computation. The execution times of **TagGen**, **ProofGen**, **Verifying** phases are shown in Fig. 3 to Fig. 5, respectively. The **TagGen** time of our scheme is linear with the number $l$ of data blocks, and the execution times are 2.296 s and 5.742 s, whereas the execution times are 4.593 s and11.484 s while in Zhang et al.'s scheme when $l$=100 and $l$=250 . In the **ProofGen phase,** time cost mostly comes from the point multiplication operation in $G$, and the execution time is 0.512 s and 1.281 s in our PIDPA, whereas the execution times are 1.025 s and 2.563 s in Zhang et al.'s scheme when $l$=100 and $l$=250, respectively. In **Verifying** phase, the execution times are 0.549 s and 1.318 s in our PIDPA, and Zhang et a.'s scheme needs 0.555 s and 1.324 s when $l$=100 and $l$=250, respectively. The time gap between the two schemes stem from a pairing operation and a point multiplication operation $O_p - O_M$.

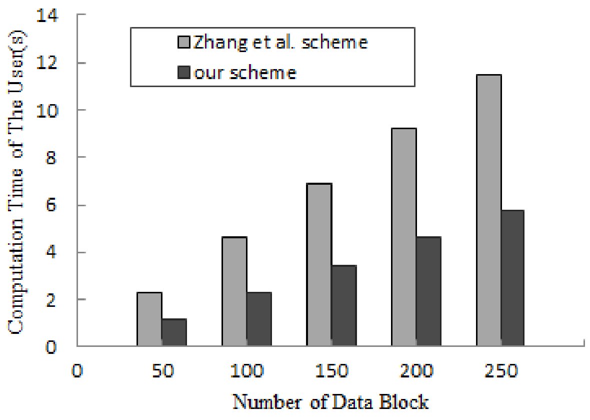

**Fig. 3.** Comparison of computation cost of the data user

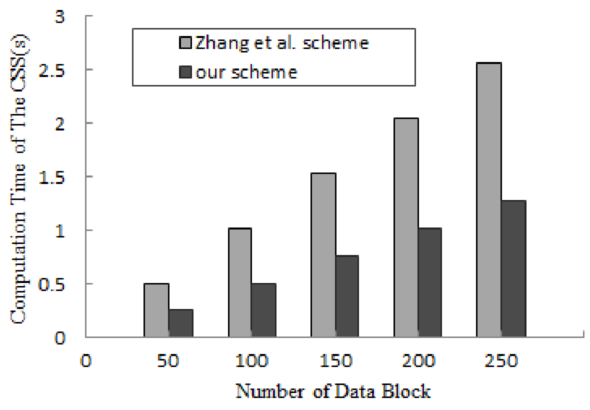

**Fig. 4.** Comparison of computation cost of the CSS

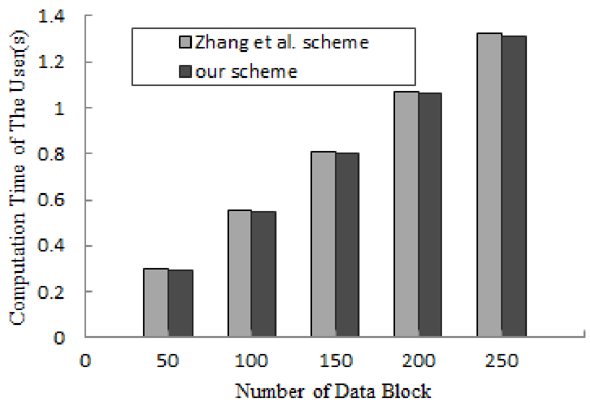

**Fig. 5.** Comparison of computation cost of the TPA

As batch auditing is supported in the multi-user scenario of our PIDPA scheme, the computation cost of the TPA is restrained by the number $l$ of data blocks and the number $k$ of data users. The relation diagram of computation cost with $l$ and $k$ is shown in Fig. 6.

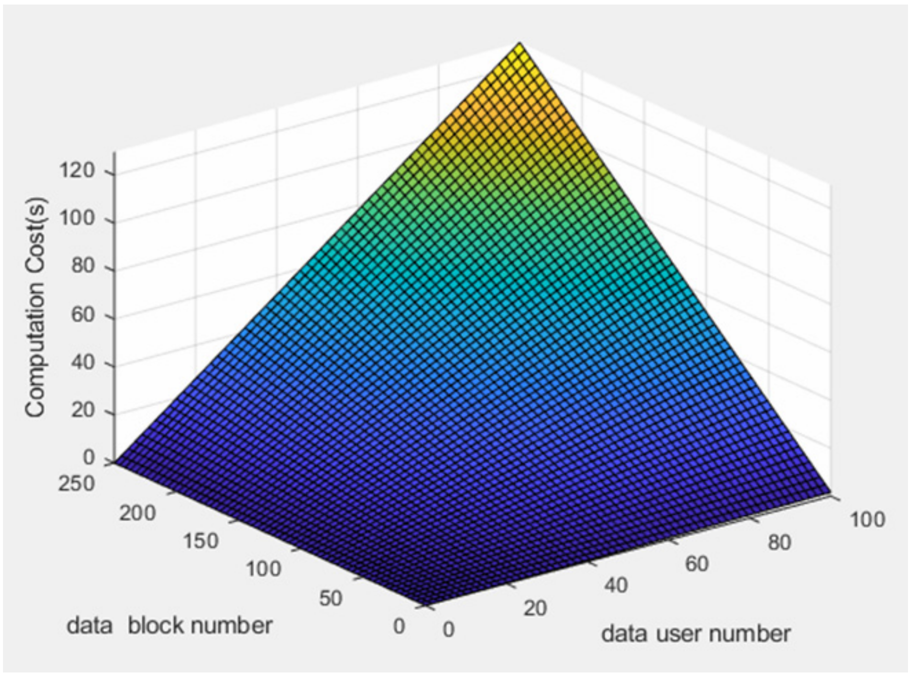

**Fig. 6.** Computation cost of the TPA with different number of the data blocks and the number of data users

## 7 Conclusion

In this study, we propose the first pseudo-ID-based public auditing scheme that is proven secure on the assumption that the CDH problem. Our PIDPA is successfully implemented to protect the identity privacy of data users and the privacy of outsourced data, which prevents the TPA from obtaining any information of the data user or data block during auditing. In the multi-user scenario, batch auditing is supported and the certificate management key escrow problems are avoided. Our security analysis demonstrates that the proposed PIDPA scheme satisfies all the security requirements for data auditing in the cloud. Compared with previous schemes, our proposed scheme performs better than Zhang et al.'s scheme and can address the security problems in that scheme.

As part of our future work, we aim to analyze in detail the security of the scheme proposed in this paper and design an improved security model to enhance the security features. Moreover, we will design an efficient public auditing scheme for outsourced data with constant verification time, high security, and optimal performance to meet practical requirements.

## Acknowledgments

The authors wish to thank the anonymous referees for their patience in reading this manuscript and their invaluable comments and suggestions.

The work was supported by the National Natural Science Foundation of China under grant (61662071, 61562077).

## References

[1] X. Li, S. Kumari, J. Shen, F. Wu, C. Chen, S.H. Islam, Secure Data Access and Sharing Scheme for Cloud Storage, Wireless Personal Communications 96(4)(2016) 1-20.

[2] X. Jiang, J. Yu, J. Yan, R. Hao, Enabling efficient and verifiable multi-keyword ranked search over encrypted cloud data, Information Sciences 403-404(2017) 22-41.

[3] Y. Yu, L. Xue, M.H. Au, .W. Susilo, J. Ni, Y. Zhang, A.V. Vasilakos, J. Shen, Cloud data integrity checking with an identity-based auditing mechanism from RSA, Future Generation Computer Systems 62(C)(2016) 85-91.

[4] J. Zhang, Q. Dong, Efficient ID-based public auditing for the outsourced data in cloud storage, Information Sciences 343-344(2016) 1-14 DOI:10.1016/j.ins.2015.12.043

[5] S. Yu, Big privacy: challenges and opportunities of privacy study in the age of big data, IEEE Access 4(2016) 2751-2763.

[6] T. Ma, J. Zhou, M. Tang, Y. Tian, A. AI-Dhelaan, M. AI-Rodhaan, S. Lee, Social Network and Tag Sources Based Augmenting Collaborative Recommender System, IEICE Transactions on Information & Systems 98(4)(2015) 902-910.

[7] M. Ali, S.U. Khan, A.V. Vasilakos, Security in cloud computing: opportunities and challenges, Information Sciences 305(2015) 357-383.

[8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, D. Song, Provable data possession at untrusted stores, in Proc. ACM Conference on Computer and Communications Security, 2007.

[9] H. Shacham, B. Waters, Compact proofs of retrievability, in: Proc. International Conference on the Theory and Application of Cryptology and Information Security, 2008.

[10] B. Dan, B. Lynn, H. Shacham, Short signatures from the Weil Pairing, in: Proc. International Conference on the Theory and Application of Cryptology and Information Security, 2001.

[11] E.C. Chang, J. Xu, Remote integrity check with dishonest storage server, in: Proc. European Symposium on Research in Computer Security: Computer Security, 2008.

[12] F. Sebé, J. Domingo-Ferrer, A. Martínez-Ballesté, Y. Deswarte, J.-J. Quisquater, Efficient remote data possession checking in critical information infrastructures, IEEE Transactions on Knowledge & Data Engineering 20(8)(2008) 1034-1038.

[13] Z. Xu, L. Wu, M.K. Khan, K.-K.R. Choo, D. He, A secure and efficient public auditing scheme using RSA algorithm for cloud storage, Journal of Supercomputing 73(12)(2017) 1-25.

[14] J. He, Y. Zhang, G. Huang, Y. Shi, J. Gao, Distributed data possession checking for securing multiple replicas in geographically-dispersed clouds, Journal of Computer & System Sciences 78(5)(2012) 1345-1358.

[15] 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 & Distributed Systems 22(5)(2011) 847-859.

[16] J. Zhao, C. Xu, F. Li, W. Zhang, Identity-based public verification with privacy- preserving for data storage security in cloud computing, IEICE trans. Fundam. Electron. Commun. Comput. Sci. 96(12)(2013) 2709-2716

[17] B. Wang, B. Li, H. Li, F. Li, Certificateless public auditing for data integrity in the cloud, in: Proc. Communications and Network Security. IEEE, (2013) 136-144.

[18] B. Wang, B. Li, H. Li, Knox: privacy-preserving auditing for shared data with large groups in the cloud, in: Proc. International Conference on Applied Cryptography and Network Security, 2012.

[19] B. Wang, B. Li, H. Li, Oruta: privacy-preserving public auditing for shared data in the cloud, IEEE Transactions on Cloud Computing 2(1)(2014) 43-56.

[20] H. Wang, Q. Wu, b.b. Qin, J. Domingo-Ferrer, Identity-based remote data possession checking in public clouds, IET Information Security 8(2)(2014) 114-121. DOI:10.1049/iet-ifs.2012.0271

[21] B. Wang, B. Li, H. Li, Panda: public auditing for shared data with efficient user revocation in the cloud, IEEE Transactions on Services Computing 8(1)(2015) 92-106.

[22] A. Shamir, Identity-based cryptosystems and signature schemes, in: Proc. Adv. Cryptol.-ASLACRYPT, 1985.

[23] S. Tan, Y. Jia, NaEPASC: a novel and efficient public auditing scheme for cloud data, Frontiers of Information Technology & Electronic Engineering 15(9)(2014) 794-804.

[24] L. Wu, J. Wang, D. He, M.-K. Khan, Cryptanalysis of an identity-based public auditing protocol for cloud storage, Frontiers of Information Technology & Electronic Engineering 18(12)(2017) 1972-1977.

[25] D. He, H. Wang, J. Zhang, L. Wang, Insecurity of an identity-based public auditing protocol for the outsourced data in cloud storage, Information Sciences 375(2017) 48-53.

[26] Stanford University, Applied Cryptography Group. <http://crypto.stanford.edu.>.