Multimed Tools Appl (2017) 76:19411–19428
DOI 10.1007/s11042-015-3139-7

Cryptanalysis and improvement of Panda - public auditing for shared data in cloud and internet of things

Tonghao Yang[1] · **Bin Yu**[1] · **Hengjun Wang**[1] · **Junquan Li**[1] · **Zhihan Lv**[2]

Received: 27 September 2015 / Revised: 26 October 2015 / Accepted: 30 November 2015 /
Published online: 8 December 2015

Abstract Cloud computing and internet of things have gained remarkable popularity by a wide spectrum of users recently. Despite of the convenience of cloud storage, security challenges have risen upon the fact that users do not physically possess their data any more. Thus, some auditing schemes are introduced to ensure integrity of the outsourced data. And among them Panda is a public auditing scheme for shared data with efficient and secure user revocation proposed by Wang et al. It argued that it could verify the integrity of shared data with storage correctness and public auditing. In this paper, we analyze this scheme and find some security drawbacks. Firstly, Panda cannot preserve shared data privacy in cloud storage. Furthermore, our analysis shows that Panda is vulnerable to integrity forgery attack, which can be performed by malicious cloud servers to forge a valid auditing proof against any auditing challenge even without correct data storage. Then we pinpoint that the primary cause of the insecurity is the linear combinations of sampled data blocks without random masking properly. Finally, we propose an improvement of Panda together with data privacy preserving and sound public auditing while incurring optimal communication and computation overhead.

✉ Tonghao Yang
youngtonghao@163.com

✉ Zhihan Lv
webvr@vip.qq.com

Bin Yu
byu2009@163.com

Hengjun Wang
wanghengjun@163.com

Junquan Li
junquanli2014@163.com

[1] Zhengzhou Institute of Information Science and Technology, Zhengzhou, People's Republic of China

[2] Shenzhen Institutes of Advanced Technology, Chinese Academy of Science, Shenzhen, China

Keywords Multimedia data · Cloud computing · Shared data · Privacy preserving · Public auditing · Internet of things

1 Introduction

Cloud computing and internet of things have been envisioned as a next generation information technology paradigm for provisioning of computing and storage resources with a reduced cost and fast accessibility [1]. However, its benefits in terms of flexibility are shadowed by security challenges which inhibit its adoption. In the cloud and internet of things, users put large data files on the cloud storage server. As the users' data do not reside within their physical possession any more, how to efficiently audit the integrity of outsourced data becomes a great challenge in the cloud [35]. Traditional data auditing needs data to be downloaded to the local storage, which could seriously increase the communication and computation overhead. So this is not suitable for cloud environments and internet of things because of the huge amount of users' data in the cloud and internet of things.

Many schemes including private auditing [15] and public auditing [2, 38, 42] are proposed to process data integrity checking in cloud computing. Compared with private auditing schemes, public auditing schemes allow any public verifier to check the integrity of data storage. Besides, users themselves do not have to afford the overhead of data integrity checking. Thus, public auditing seems more practical and may play a more important role in the cloud [34]. Most of the above schemes focus on the public auditing of personal data in the cloud. However, several other schemes [27, 29] concerning on the integrity checking of shared data have been proposed recently. Different blocks in shared data in the cloud are signed by different users during the data modification. Moreover, users who leave the group or misbehave must be revoked from the group for security consideration. Revoked users cannot access or modify shared data any more. Furthermore, the data blocks previously signed by the revoked users still need to be re-signed by an existing user in the group though the content of shared data is not changed during user revocation.

However, none of the above schemes considers the efficiency of user revocation when checking the integrity of shared data. And then, Panda [32] and its variants [28, 30, 31] are proposed to solve this problem. According to [32], Panda is able to allow cloud server to efficiently re-sign data blocks on behalf of existing users in the group during user revocation, thus existing users do not need to download and re-sign multimedia data blocks by themselves. Moreover, a public verifier is able to check the integrity of shared data without retrieving the entire data blocks from the cloud. Unfortunately, we find that Panda and its variants can neither preserve data privacy nor resist against integrity forgery attack. These schemes are insecure and vulnerable to attacks from outside attackers or malicious servers.

- Contribution. In this paper, we make a cryptanalysis of Panda and show two specific attacks on Panda: data recovery attack can be implemented by outside attackers to reveal data privacy; while integrity forgery attack can be performed by malicious cloud servers to forge an auditing proof against any auditing challenge successfully even without correct data storage. We pinpoint that the primary cause of the insecurity is the linear combinations of sampled data blocks without random masking properly. Then we propose an improvement of Panda together with data privacy preserving and sound public auditing while incurring optimal communication and computation overhead. The cryptanalysis and improvement are also available for Panda's variants.
- Organization. The rest of the paper is organized as follows. Section II presents the problem statement of Panda. Then a brief description of Panda is given in Section III. In Section VI,

we introduce two attacks on Panda. An improvement of Panda is proposed in Section V and then the security analysis of the new proposal is followed in Section VI. Performance evaluation is given in Section VII.

1.1 Problem statement

The system model, design goals, several preliminaries of Panda and the improved scheme are described in this section.

1.2 System and security model

The system comprises three different entities: the cloud, the users (who share data as a group), and the public verifier, as shown in Fig. 1.

- The cloud. The cloud owns the infrastructure and expertise to offer outsourced data storage and sharing services to the group.
- The users. The users can be classified into two types, original users and other group users. The original users create and share data with other group users in the cloud. Both the original users and group users can modify shared data.
- The public verifier. A client or a third-party auditor (TPA), who can provide data integrity verification, aims to check the storage reliability and validity of shared data via a challenge-response protocol with the cloud server.

Users can put large data files on the cloud and internet of things to free themselves from the burden of storage and maintenance. Shared data is divided into several blocks. Users in the group can perform insert, delete and update operations on the blocks. Each block in shared data is attached with a signature. The original user creates all the signatures on shared data initially. After that, users who modify data blocks are required to sign the modified block with their own private key. Assume that the original user is the group manager and can revoke users

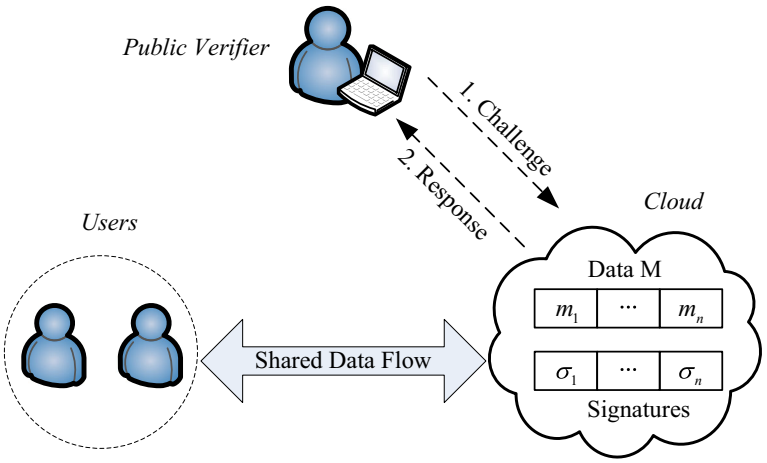


Fig. 1 The system model of Panda

on behalf of the group. Once users are revoked from the group, the blocks previously signed by the revoked users need to be re-signed by an existing user.

1.3 Design goals

Panda and the improved scheme are required to achieve the following security and performance goals:

- Storage Correctness. The public verifier can audit the integrity of shared data correctly.
- Efficient and Secure User Revocation. The data blocks signed by revoked users can be efficiently re-signed and revoked users cannot create valid signatures any more.
- Public Auditing. The public verifier can check the integrity of shared data without retrieving the entire data from the cloud.
- Scalability. Data can be efficiently shared among a large number of users in the cloud, and the public verifier can manage multiple auditing tasks from possibly many users concurrently in secure and efficient manner.
- Privacy preserving. The public verifier cannot derive the content of shared data from information collected during integrity checking. for the performing of public auditing between cloud and public verifier will reveal data privacy to outside attackers in Panda, this design goal is only achieved by the improved scheme.

1.4 Preliminaries

Some necessary cryptographic primitives are introduced as follows.

- Bilinear Maps. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_1$. Bilinear map e is a map e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties: 1) Bilinear: for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$; 2) Non-degenerate: $e(g, g) \neq 1$; 3) Computable: there exists an efficient algorithm for computing map e.
- Discrete Logarithm (DL) Problem. Let $a \in \mathbb{Z}_p^*$, given $g, g^a \in \mathbb{G}_1$ as input, output a.
- Computational Diffie-Hellman (CDH) Problem. Let $a, b \in \mathbb{Z}_p^*$, given $g, g^a, g^b \in \mathbb{G}_1$ as input, output $g^{ab} \in \mathbb{G}_1$.
- Homomorphic Authenticators. Homomorphic authenticators are used to construct public auditing mechanisms. Homomorphic authenticable signature schemes should satisfy the following properties: unforgeability, blockless verifiability, and non-malleability [25].
- Proxy Re-signatures. A semi-trusted proxy is able to act as a translator of signatures between two users. Specifically, the proxy can translate one user's signature into the other's on the same block without learning any private information of the two users [3]. In Panda, the cloud is the proxy and translates signatures during user revocation.

2 Panda description

Panda uses public key-based homomorphic authenticators, which are based on the BLS signature scheme [4], to equip the auditing with public auditability. Proxy re-signatures are

also used to support cloud server to re-sign shared data blocks. The security of Panda is based on the hardness assumptions of CDH and DL problem over bilinear groups. This section introduces the construction of Panda and its extension.

2.1 Construction of Panda

Details of Panda construction are illustrated in Fig. 2.

At last, if and only if the verification result is true, the public verifier believes that the integrity of all the blocks in shared data is correct.

2.2 Extension of Panda

Panda can be extended in terms of detection probability, scalability, and reliability. Thus, the detection probability is improved by performing multiple auditing tasks on the same shared data; batch auditing is supported by verifying multiple auditing tasks simultaneously; and reliability is enhanced by adoption of a multi-proxy model of Panda. Further details of Panda extension can be found in [32].

3 Attacks on Panda

In fact, not only is the cloud semi-trusted, but also the public verifier is not fully trusted. Pubic auditing allows any potential client to verify the integrity of the cloud data. As described in Panda, a client who wants to use the shared data in cloud can act as the public verifier. So the public verifier may collect the auditing information for his own purpose (e.g., revealing data privacy, etc.)

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of prime order p, g be a generator of $\mathbb{G}_1$, $e: \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear map, w be another generator of $\mathbb{G}_1$. H is a hash function with $H: \{0,1\}^* \to \mathbb{G}_1$. n is the total number of blocks in shared data which is described as $M = (m_1, m_2, \cdots, m_n)$. d is the total number of users in the group. Panda includes six algorithms: KeyGen, ReKey, Sign, ReSign, ProofGen, and ProofVerify.

•**KeyGen.** User u_i generates a random $\pi_i \in \mathbb{Z}_p^*$, and outputs public key $pk_i = g^{\pi_i}$ and private key $sk_i = \pi_i$.

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

1) the cloud generates a random $r \in \mathbb{Z}_p$ and sends it to user u_i.

2) user u_i sends r/π_i to user u_j, where $sk_i = \pi_i$.

3) user u_j sends $r\pi_j/\pi_i$ to the cloud, where $sk_j = \pi_j$.

4) the cloud recovers $rk_{i\to j} = \pi_j/\pi_i$.

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

•**ReSign.** Given re-signing key $rk_{i\to j}$, public key pk_i, signature σ_k, block m_k and block identifier id_k, the cloud first checks that if $e(\sigma_k, g) = e(H(id_k)w^{m_k}, pk_i)$. If the verification result is 0, the cloud outputs $\perp$; otherwise, it outputs

$$\sigma'_k = \sigma_k^{rk_{i\to j}} = (H(id_k)w^{m_k})^{\pi_i \pi_j/\pi_i} = (H(id_k)w^{m_k})^{\pi_j}.$$

•**ProofGen.** A public verifier generates an auditing challenge as follows:

1) Randomly picks a c-element subset L of set $[1, n]$ to locate the c selected random blocks that will be checked in this auditing task.

2) Generates a random $\eta_l \in \mathbb{Z}_q^*$, for $l \in L$ and q is a much smaller prime than p.

3) Outputs an auditing challenge $\{(l, \eta_l)\}_{l\in L}$, and sends it to the cloud.

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

1) According to the signer identifier of each selected block, the cloud divides set L into d subsets $L_1, L_2, \cdots, L_d$, where L_i is the subset of selected blocks signed by user u_i. The number of elements in subset L_i is c_i. Clearly, we have $c = \sum_{i=1}^{d} c_i$, $L = L_1 \cup L_2 \cup \cdots \cup L_d$ and $L_i \cap L_j = \varnothing$, for $i \neq j$.

2) For each subset L_i, the cloud computes $\alpha_i = \sum_{l\in L_i} \eta_l m_l$ and $\beta_i = \prod_{l\in L_i} \sigma_l^{\eta_l}$.

3) The cloud outputs an auditing proof $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, id_l\}$, and sends it to the verifier, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_d)$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_d)$.

•**ProofVerify.** Given an auditing challenge $\{(l, \eta_l)\}_{l\in L}$, an auditing proof $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, id_l\}$, and all the existing users' public keys $(pk_1, pk_2, \cdots, pk_d)$, the public verifier checks the correctness of this auditing proof as if

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

Fig. 2 Details of Panda

It is claimed in [32] that Panda can achieve the following goals: storage correctness, efficient and secure user revocation, public auditing, and scalability. Unfortunately, we find that Panda is vulnerable to attacks from outside attackers and malicious cloud servers. More concretely, the performing of public auditing between cloud and public verifier will reveal data privacy to outside attackers; and the storage correctness of shared data in the cloud may not be ensured even if the cloud passes the integrity auditing from public verifier.

We first describe the threat model of Panda in this section. Then, we introduce two specific attacks on Panda. Data recovery attack can be performed by outside attackers to reveal data privacy. Integrity forgery attack can be implemented by malicious cloud servers to forge an auditing proof against any auditing challenge successfully. Note that the two attacks are also available on panda's variants [28, 30, 31].

3.1 Threat model

- Privacy Threats. The content of shared data should be private. An outside attacker acting as a public verifier, who is only allowed to verify the correctness of shared data integrity, may try to recover the content of data blocks. Once the outside attacker reveals the content directly or indirectly, he finishes the privacy attack successfully.
- Integrity Threats. The cloud server may corrupt or even delete the shared data in cloud storage because of software bugs or hardware failures inadvertently. Besides, the cloud server can be self-interested. It may be economically motivated, which means it might even hide these data corruption incidents to cloud users in order to save its reputation and avoid profit loss of its services.

3.2 Data recovery attack

We assume that an outside attacker who wants to reveal the content of shared data acts as a public verifier. And the outside attacker performs public auditing process with the cloud server. We will show that he can achieve this goal after performing enough public auditing process. The detailed attack scheme is as follows.

For the sake of simplicity, the outside attacker firstly targets only at data blocks signed (or re-signed) by user u_1. For further attacks targeting at data blocks signed (or re-signed) by user $u_i, 2 \le i \le d$, the attack scheme below is also available.

We assume that the number of elements in subset L_1 is $c_1 = t$ and $L_1 = \{l_1, l_2, \cdots, l_t\}$. After the user and the cloud finish KeyGen, ReKey (optional), Sign and ReSign (optional), the outside attacker perform t times of ProofGen process and the auditing challenges are $ch_1 = \{(l_1, \eta_{1l_1}), (l_2, \eta_{1l_2}), \cdots, (l_t, \eta_{1l_t})\}$, $ch_2 = \{(l_1, \eta_{2l_1}), (l_2, \eta_{2l_2}), \cdots, (l_t, \eta_{2l_t})\}$, …, $\{ch_t = \{(l_1, \eta_{tl_1}), (l_2, \eta_{tl_2}), \cdots, (l_t, \eta_{tl_t})\}$. Then he sends them to the cloud and receives auditing proofs as $P_1 = (\boldsymbol{\alpha}_1, \boldsymbol{\beta}_1, id_l)$, $P_2 = (\boldsymbol{\alpha}_2, \boldsymbol{\beta}_2, id_l)$, …, $P_t = (\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t, id_l)$.

Since the attack firstly targets only at data blocks signed (or re-signed) by user u_1, there is only one element in vector $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ respectively, which is $\alpha_i = \sum_{j=1}^{t} \eta_{il_j} m_{l_j}, 1 \le i \le t$ and $\beta_i = \prod_{j=1}^{t} \sigma_{l_j}{}^{\eta_{il_j}}, 1 \le i \le t$, where $\sigma_{l_j} = \left(H(id_{l_j}) w^{m_{l_j}}\right)^{\pi_1}, 1 \le j \le t$.

Assume that $\boldsymbol{\eta}_1 = (\eta_{1l_1}, \eta_{1l_2}, \cdots, \eta_{1l_t})$, $\boldsymbol{\eta}_2 = (\eta_{2l_1}, \eta_{2l_2}, \cdots, \eta_{2l_t})$, …, $\boldsymbol{\eta}_t = (\eta_{tl_1}, \eta_{tl_2}, \cdots, \eta_{tl_t})$ and the construction of matrix $\boldsymbol{\eta}^1$ is

$$\boldsymbol{\eta}^1 = \begin{bmatrix} \eta_{1l_1} & \eta_{1l_2} & \cdots & \eta_{1l_t} \\ \eta_{2l_1} & \eta_{2l_2} & \cdots & \eta_{2l_t} \\ \vdots & \vdots & \vdots & \vdots \\ \eta_{tl_1} & \eta_{tl_2} & \cdots & \eta_{tl_t} \end{bmatrix}$$

Let $\det(\boldsymbol{\eta}^1)\neq 0$, so vectors $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \cdots, \boldsymbol{\eta}_t$ are linearly independent. Then there is a matrix $\boldsymbol{\mu}$ that satisfies $\boldsymbol{\mu}\boldsymbol{\eta}^1 = \mathbf{E}$.

Assume that matrix $\mathbf{m}_1 = [m_{l_1} \quad m_{l_2} \quad \cdots \quad m_{l_t}]$ is constructed from data blocks signed (or re-signed) by user u_1 and $\boldsymbol{\alpha}' = [\alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_t]$. Thus $\boldsymbol{\alpha}' = \boldsymbol{\eta}^1\mathbf{m}_1$, and then the outside attacker can derive $\mathbf{m}_1 = \boldsymbol{\mu}\boldsymbol{\alpha}'$. The outside attacker recovers the data blocks signed (or re-signed) by user u_1 successfully.

In fact, even if the outside attacker cannot act as the public verifier in some cases, if he eavesdrops on enough auditing challenges $\{(l,\eta_l)\}_{l\in L}$ and auditing proofs $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, id_l\}$, he can also recover the matrix $\mathbf{m}_i$, namely the data blocks signed (or re-signed) by user $u_i, 1\leq i\leq d$. And the exact number of pairs of auditing challenges and proofs he need to collect is c_i that satisfies $\det(\boldsymbol{\eta}^i)\neq 0, 1\leq i\leq d$.

3.3 Integrity forgery attack

We assume that malicious cloud servers might delete data owned by users or even hide some data corruptions for their own benefits. We will show that a malicious cloud server can forge an auditing proof against any auditing challenge successfully even without correct data storage.

To make matters worse, according to the following attack scheme, an outside attacker, who does not initially possess the shared data at all, can forge an auditing proof against any auditing challenge after eavesdropping on enough valid pairs of auditing challenges and auditing proofs. This means that any user is able to masquerade as the cloud server as long as he can eavesdrop on auditing challenges and auditing proofs. This serious security flaw can bring unexpected risks to both users and the cloud. The detailed scheme of integrity forgery attack is as follows.

For the sake of simplicity, the attacker firstly targets only at data blocks signed (or re-signed) by user u_1. For further attacks targeting at data blocks signed (or re-signed) by user u_i, $2\leq i\leq d$, the attack scheme below is also available.

As the same as data recovery attack, we assume that the number of elements in subset L_1 is $c_1 = t$ and $L_1 = \{l_1, l_2, \cdots, l_t\}$. The user and the malicious cloud server have finished KeyGen, ReKey (optional), Sign and ReSign (optional), After the public verifier performs t times of ProofGen process and the auditing challenges are $ch_1 = \{(l_1, \eta_{1l_1}), (l_2, \eta_{1l_2}), \cdots, (l_t, \eta_{1l_t})\}$, $ch_2 = \{(l_1, \eta_{2l_1}), (l_2, \eta_{2l_2}), \cdots, (l_t, \eta_{2l_t})\}$, …, $\{ch_t = \{(l_1, \eta_{tl_1}), (l_2, \eta_{tl_2}), \cdots, (l_t, \eta_{tl_t})\}$, the malicious cloud server returns auditing proofs as $P_1 = (\boldsymbol{\alpha}_1, \boldsymbol{\beta}_1, id_l)$, $P_2 = (\boldsymbol{\alpha}_2, \boldsymbol{\beta}_2, id_l)$, …, $P_t = (\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t, id_l)$.

Since the attack firstly targets only at data blocks signed (or re-signed) by user u_1, there exists only one element in vector $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ respectively, which is $\alpha_i = \sum_{j=1}^{t} \eta_{il_j} m_{l_j}, 1\leq i\leq t$ and $\beta_i = \prod_{j=1}^{t} \sigma_{l_j}^{\eta_{il_j}}, 1\leq i\leq t$, where $\sigma_{l_j} = \left(H(id_{l_j}) w^{m_{l_j}}\right)^{\pi_1}, 1\leq j\leq t$.

Assume that $\boldsymbol{\eta}_1 = (\eta_{1l_1}, \eta_{1l_2}, \cdots, \eta_{1l_t})$, $\boldsymbol{\eta}_2 = (\eta_{2l_1}, \eta_{2l_2}, \cdots, \eta_{2l_t})$, …, $\boldsymbol{\eta}_t = (\eta_{tl_1}, \eta_{tl_2}, \cdots, \eta_{tl_t})$ and the construction of matrix $\boldsymbol{\eta}^1$ is

$$\boldsymbol{\eta}^1 = \begin{bmatrix} \eta_{1l_1} & \eta_{1l_2} & \cdots & \eta_{1l_t} \\ \eta_{2l_1} & \eta_{2l_2} & \cdots & \eta_{2l_t} \\ \vdots & \vdots & \vdots & \vdots \\ \eta_{tl_1} & \eta_{tl_2} & \cdots & \eta_{tl_t} \end{bmatrix}$$

If $\det(\boldsymbol{\eta}^1)\neq 0$, vectors $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \cdots, \boldsymbol{\eta}_t$ are linearly independent. Therefore, the malicious cloud server deletes data blocks signed (or re-signed) by user u_1 and stores the t pairs of auditing challenges and proofs. And then he can generate valid auditing proofs against auditing challenge to data blocks signed (or re-signed) by user u_1 even without correct data storage of them.

After receiving an auditing challenge $ch^* = \{(l_1, \eta_{l_1}{}^*), (l_2, \eta_{l_2}{}^*), \cdots, (l_t, \eta_{l_t}{}^*)\}$ to data blocks signed (or re-signed) by user u_1, the malicious cloud server generates an auditing proof as follows:

1) Assume $\boldsymbol{\eta}^* = (\eta_{l_1}{}^*, \eta_{l_2}{}^*, \cdots, \eta_{l_t}{}^*)$, since $\det(\boldsymbol{\eta}^1)\neq 0$, the malicious cloud server can generate $a_i, 1\leq i\leq t$ that satisfies $\boldsymbol{\eta}^* = a_1\boldsymbol{\eta}_1 + a_2\boldsymbol{\eta}_2 + \cdots + a_t\boldsymbol{\eta}_t$, namely $\eta_{l_j}{}^* = \sum_{i=1}^{t} a_i \eta_{il_j}$.
2) The malicious cloud server generates $\alpha^* = \sum_{i=1}^{t} a_i\alpha_i$ and $\beta^* = \prod_{i=1}^{t} \beta_i{}^{a_i}$, and outputs an auditing proof $\{\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, id_l\}$, where $\boldsymbol{\alpha}^* = (\alpha^*)$ and $\boldsymbol{\beta}^* = (\beta^*)$.

Thus, when the integrity forgery attack firstly targets at data blocks signed (or re-signed) by user u_1, namely for the $i=1$ factors in $e\left(\prod_{i=1}^{d}\beta_i, g\right) = \prod_{i=1}^{d} e\left(\prod_{l\in L_i} H(id_l)^{\eta_l} \cdot w^{\alpha_i}, pk_i\right)$, the auditing proof $\{\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, id_l\}$ generated by the malicious cloud server can pass the verification because of the following equation:

$$\begin{aligned}
e(\beta^*, g) &= e\left(\prod_{i=1}^{t}\beta_i{}^{a_i}, g\right) \\
&= e\left(\prod_{i=1}^{t}\prod_{j=1}^{t}\sigma_{l_j}{}^{\eta_{il_j}a_i}, g\right) \\
&= e\left(\prod_{j=1}^{t}(\sigma_{lj})^{\sum_{i=1}^{t} a_i\eta_{il_j}}, g\right) \\
&= e\left(\prod_{j=1}^{t}\left(H(id_{l_j})w^{m_{l_j}}\right)^{\pi_1\sum_{i=1}^{t} a_i\eta_{il_j}}, g\right) \\
&= e\left(\prod_{j=1}^{t}H(id_{l_j})^{\sum_{i=1}^{t} a_i\eta_{il_j}} \cdot w^{\sum_{i=1}^{t} a_i \sum_{j=1}^{t}\eta_{il_j}m_{l_j}}, g^{\pi_1}\right) \\
&= e\left(\prod_{j=1}^{t}H(id_{l_j})^{\sum_{i=1}^{t} a_i\eta_{il_j}\pi_1} \cdot w^{\sum_{i=1}^{t} a_i\alpha_i\pi_1}, pk_1\right) \\
&= e\left(\prod_{j=1}^{t}H(id_{l_j})^{\eta_{l_j}{}^*} \cdot w^{\alpha^*}, pk_1\right)
\end{aligned}$$

Since the auditing challenges received by the malicious cloud server are the simple arrangement of auditing challenge for each data block signed (or re-signed) by each user $u_i, 2\leq i\leq d$, it's easy to separate the auditing challenges signed (or re-signed) by different users. The auditing proofs can be separated in the same way. Thus the malicious cloud server can get pairs of auditing challenges and auditing proofs for data blocks signed (or re-signed) by each user $u_i, 2\leq i\leq d$. When receiving auditing challenge, the malicious cloud server firstly generates the partial auditing proof $\{\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, id_l\}$ for each $i, 2\leq i\leq d$ specialized

in the auditing challenge, and then he gets the valid auditing proof by arranging the partial auditing proof in the way of the auditing challenge arrangement.

Thus even without correct data storage, the malicious cloud server can forge an auditing proof against any auditing challenge successfully.

4 New proposal for Panda

In this section, we propose an improved scheme of Panda, which is also a homomorphic authenticable proxy re-signature scheme. The system model and security preliminaries of this scheme are the same as Panda's. However, because Panda cannot preserve data privacy during auditing process, we add privacy preserving as the new design goal of the improved scheme to ensure that the public verifier cannot derive shared data during integrity auditing.

From the two specific attacks on Panda described above, it is concluded that the primary cause of the insecurity of Panda is the linear combinations of sampled data blocks without random masking properly. Data blindness during data auditing is not well concerned in Panda. Thus outside attackers and malicious cloud servers can easily derive shared data content or forge a valid auditing proof by collecting enough linear combinations of data blocks. So in the improved scheme, we integrate the homomorphic authenticators with random masking technique.

Actually, the data privacy preserving in public auditing schemes and its solution with random masking have already been discussed and proposed (Wang et al., 2010). But random masking is not properly used in (Wang et al., 2010). In fact, this solution is found of security flaws and cannot provide secure data storage for users [39]. However, in our new proposal with improved random masking technique, the outside attackers and malicious cloud servers cannot get necessary information to derive the shared data content or generate valid auditing proofs any more, no matter how many linear combinations of data blocks can be collected. And even with the presence of the randomness, the correctness validation of the pairs of auditing challenges and proofs can still be processed in a new way.

The new proposal for Panda also includes the following six algorithms: KeyGen, ReKey, Sign, ReSign, ProofGen, and ProofVerify. The initialization and the first four algorithms of the new proposal are the same as Panda. The ProofGen and ProofVerify algorithms are illustrated in Fig. 3.

At last, if and only if the verification result is true, the public verifier believes that the integrity of all the blocks in shared data is correct.

The proof of the correctness of the verification is given as follows.

$$
\begin{aligned}
& e\left(\prod_{i=1}^{d}\beta_i, g\right) \\
&= \prod_{i=1}^{d} e\left(\prod_{l\in L_i}\sigma_l{}^{\eta_l}, g\right) \\
&= \prod_{i=1}^{d} e\left(\prod_{l\in L_i}\left(H(id_l)w^{m_l}\right)^{\pi_i\eta_l}, g\right) \\
&= \prod_{i=1}^{d} e\left(\prod_{l\in L_i}H(id_l)^{\eta_l}\cdot\prod_{l\in L_i}w^{m_l\eta_l}, g^{\pi_i}\right) \\
&= \prod_{i=1}^{d} e\left(\prod_{l\in L_i}H(id_l)^{\eta_l}\cdot w^{\sum_{l\in L_i}\eta_l m_l+\gamma_i}\cdot w^{-\gamma_i}, pk_i\right) \\
&= \prod_{i=1}^{d} e\left(\prod_{l\in L_i}H(id_l)^{\eta_l}\cdot w^{\alpha_i}\cdot R_i^{-1}, pk_i\right)
\end{aligned}
$$

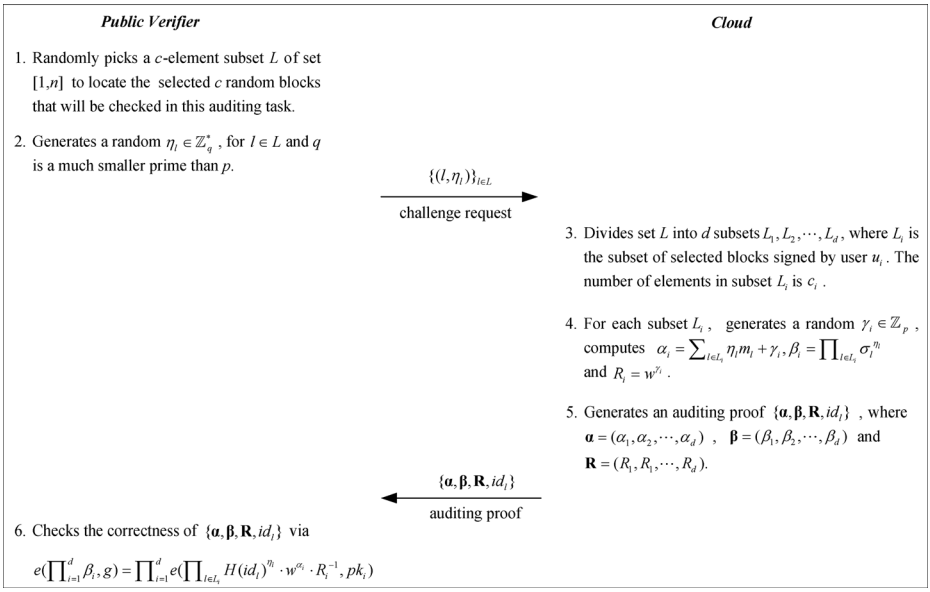


Fig. 3 The improvement of Panda

As the same as Panda, this new proposal can also be extended in terms of detection probability, scalability, and reliability easily. Due to space limitation, we will not describe this extension and details of the extended construction can refer to [32].

5 Security analysis

We evaluate the security of the new proposal for Panda by modularizing it into two parts, namely the storage correctness guarantee and the privacy preserving guarantee. The security of our scheme depends on the hardness assumption of CDH and DL problems.

- Storage Correctness Guarantee. We need to prove that the cloud server cannot generate valid auditing proof for the public verifier without actually storing the shared data.

Theorem 1. The cloud passes the verification done by the public verifier only if it indeed possesses the specified shared data intact as it is.

Proof. First, the signature scheme of the new proposal is existentially unforgeable and please refers to [4, 34]. Then, the proof follows from Theorem 4.2 of [25]. The cloud server is treated as an adversary. The challenger controls the random oracle $H(\cdot)$. We show that if the adversary passes the verification with non-negligible probability, a simulator can be constructed that can solve the CDH problem.

Given $g, g^a, g^b \in \mathbb{G}_1$, the simulator needs to output $g^{ab} \in \mathbb{G}_1$. The simulator randomly chooses $x, y \in \mathbb{Z}_p$ and then sets $pk_1 = g^a$ and $w = g^x g^{by}$. For each l, the simulator chooses $r_l \in \mathbb{Z}_p$, and programs the random oracle at l as $H(id_l) = g^{r_l - xm_l - bym_l}$.

Since $w=g^x g^{by}$, the simulator computes σ_l for the signature query issued by the adversary as

$$\sigma_l = (H(id_l)w^{m_l})^a = \left(g^{r_l - xm_l - bym_l}\left(g^x g^{by}\right)^{m_l}\right)^a = g^{ar_l}$$

Firstly, for an auditing challenge returned by the challenger, let $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{R}, id_l\}$ be the cloud server's response that can also satisfy

$$e\left(\prod_{i=1}^{d}\beta_i, g\right) = \prod_{i=1}^{d} e\left(\prod_{l\in L_i} H(id_l)^{\eta_l}\cdot w^{\alpha_i}\cdot R_i^{-1}, pk_i\right) \tag{1}$$

And then for the same $\gamma_i \in \mathbb{Z}_p$, let the adversary output $\{\boldsymbol{\alpha}', \boldsymbol{\beta}', \mathbf{R}, id_l\}$ as the auditing proof and it satisfies

$$e\left(\prod_{i=1}^{d}\beta_i', g\right) = \prod_{i=1}^{d} e\left(\prod_{l\in L_i} H(id_l)^{\eta_l}\cdot w^{\alpha_i'}\cdot R_i^{-1}, pk_i\right) \tag{2}$$

Obviously that $\alpha_i \neq \alpha_i'$, otherwise $\beta_i = \beta_i'$, which contradicts the assumption that the challenger aborted on the adversary's response. Assume that $\Delta\alpha_i = \alpha_i' - \alpha_i$, we can solve the CDH problem as follows:

Since $\gamma_i \in Z_p$ is the same in Eqs. (1) and (2), dividing Eq. (2) by Eq. (1), we have

$$e\left(\prod_{i=1}^{d}\frac{\beta_i'}{\beta_i}, g\right) = \prod_{i=1}^{d} e\left(w^{\Delta\alpha_i}, pk_i\right) \tag{3}$$

For the $i=1$ factors in Eq. (3), replacing w by $g^x g^{by}$ and pk_1 by g^a, we have

$$e\left(\beta_1'\cdot\beta_1^{-1}, g\right) = e\left(\left(g^x g^{by}\right)^{\Delta\alpha_1}, g^a\right) = e\left(pk_1{}^{x\Delta\alpha_1}\left(g^{by\Delta\alpha_1}\right)^a, g\right)$$

Thus we have

$$\begin{aligned} &e\left(pk_1{}^{-x\Delta\alpha_1}, g\right)e\left(\beta_1'\cdot\beta_1^{-1}, g\right) \\ &= e\left(pk_1{}^{-x\Delta\alpha_1}, g\right)e\left(pk_1{}^{x\Delta\alpha_1}\left(g^{by\Delta\alpha_1}\right)^a, g\right)e\left(\beta_1'\cdot\beta_1^{-1}\cdot pk_1{}^{-x\Delta\alpha_1}, g\right) = e\left(\left(g^{by\Delta\alpha_1}\right)^a, g\right) \\ &= e\left(g^{ab}, g\right)^{y\Delta\alpha_1} e\left(\left(\beta_1'\cdot\beta_1^{-1}\cdot pk_1{}^{-x\Delta\alpha_1}\right)^{1/y\Delta\alpha_1}, g\right) = e\left(g^{ab}, g\right) \end{aligned}$$

So as long as $y\Delta\alpha_1 \neq 0 \bmod p$, we can solve the CDH problem as

$$g^{ab} = \left(\beta_1'\cdot\beta_1^{-1}\cdot pk_1{}^{-x\Delta\alpha_1}\right)^{1/y\Delta\alpha_1}$$

Since the adversary cannot get the value of y, the probability that $y\Delta\alpha_1 = 0 \bmod p$ will be $1/p$ which is negligible and therefore $\beta_1 = \beta_1'$. And in this case if the adversary successes with non-negligible probability, a simulator can be constructed that can solve the DL problem.

We have proved that $\beta_1=\beta_1^{'}$. It is only the values α_i and α_i that can differ. The simulator answers the adversary's queries and the adversary outputs a forged proof $\{\boldsymbol{\alpha}', \boldsymbol{\beta}', \mathbf{R}, id_l\}$. Then we have

$$e\left(\beta_1^{'}, g\right) = e(\beta_1, g)e\left(\prod_{l\in L_1} H(id_l)^{\eta_l} \cdot w^{\alpha_1^{'}} \cdot {R_1}^{-1}, pk_1\right)$$
$$= e\left(\prod_{l\in L_1} H(id_l)^{\eta_l} \cdot w^{\alpha_1} \cdot {R_1}^{-1}, pk_1\right) e\left(w^{\alpha_1^{'}}, pk_1\right) = e(w^{\alpha_1}, pk_1) w^{\Delta\alpha_1} = 1$$

In this case, $\Delta\alpha_1=0 \bmod p$ and this contradicts our assumption. Otherwise, we can solve the DL problem as

$$1 = w^{\Delta\alpha_1} = \left(g^x g^{by}\right)^{\Delta\alpha_1} = g^{x\Delta\alpha_1} \cdot g^{by\Delta\alpha_1}$$

Then the solution for the DL problem is

$$g^b = g^{-\frac{x\Delta\alpha_1}{y\Delta\alpha_1}} = g^{-\frac{x}{y}}$$

And y is zero only with probability $1/p$, which is negligible and this completes the proof.

The analysis above shows that there is negligible probability that an adversary can cause the public verifier to accept its proof except by responding with correctly computed values.

- Privacy Preserving Guarantee. We need to prove that the public verifier cannot derive the shared data from the information collected during integrity checking.

Theorem 2. From the cloud's auditing proof $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{R}, id_l\}$, The public verifier cannot recover any block in the shared data.

Proof. First, we show that privacy of $\sum_{l\in L_i} \eta_l m_l$ is guaranteed from α_i. This is because α_i is blinded by γ_i as $\alpha_i = \sum_{l\in L_i} \eta_l m_l + \gamma_i$ where γ_i is chosen randomly by the cloud and is unknown to the public verifier. Even with $\mathbf{R}$, due to the hardness assumption of DL problem, γ_i is still hidden from the public verifier. Thus no information of $\sum_{l\in L_i} \eta_l m_l$ can be learned from α_i.

Second, we show that privacy of $\sum_{l\in L_i} \eta_l m_l$ is guaranteed from β_i.

$$\beta_i = \prod_{l\in L_i} \sigma_l{}^{\eta_l}$$
$$= \prod_{l\in L_i} (H(id_l) w^{m_l})^{\pi_i \eta_l}$$
$$= \prod_{l\in L_i} H(id_l)^{\pi_i \eta_l} \cdot \left(w^{\sum_{l\in L_i} \eta_l m_l}\right)^{\pi_i}$$

From the equation above, we can see that $\left(w^{\sum_{l\in L_i} \eta_l m_l}\right)^{\pi_i}$ is blinded by $\prod_{l\in L_i} H(id_l)^{\pi_i \eta_l}$. Computing the value of $\prod_{l\in L_i} H(id_l)^{\pi_i \eta_l}$ from $H(id_l)$, η_l and g^{π_i}, which is the only information the public verifier can get, is hard due to the CDH problem. Thus the value of $\left(w^{\sum_{l\in L_i} \eta_l m_l}\right)^{\pi_i}$ as well as $\sum_{l\in L_i} \eta_l m_l$ cannot be derived from β_i. This completes the proof.

6 Performance evaluation

In this section, we evaluate the performance of our scheme by evaluating the time and communication overhead. Then we show the performance of auditing experiments. Results show that the new proposal for Panda provides the desired data privacy preserving and sound public auditing while incurring a little extra communication and computation overhead compared with Panda.

- Communication Overhead. As the same as in [32], the scheme does not introduce communication overhead to existing users during user revocation. Thus we only analyze the communication overhead incurred by auditing challenge and its corresponding auditing proof. As we have described above, we assume c random data blocks that will be checked in auditing process. The size of an auditing message is $c \cdot(|n|+|p|)$ bits. The size of an auditing proof $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{R}, id_l\}$ is $3d|p|+c(|id|)$ bits. Thus the total communication overhead is $3d|p|+c(|id|+|n|+|p|)$ bits. Compared with Panda, the extra communication overhead of this new proposal is only $d|p|$. Moreover, since the scheme is based on the BLS short signatures, we have the shortest auditing challenge and auditing proof which shows that the communication complexity is constant and asymptotically it is $O(1)$.
- Computation Overhead. The initialization and the first four algorithms (KeyGen, ReKey, Sign, ReSign) of the new proposal are the same as Panda. Thus the computation overhead of the first four algorithms is as the same as Panda, which can refer to [32]. Based on the auditing equation illustrated in Fig. 3, the computation overhead of an auditing proof verification is $(c+d)\ Exp_{\mathbb{G}_1}+\ (c+2d)\ Mul_{\mathbb{G}_1}+\ (d+1)\ Pair +d\,Mul_{\mathbb{G}_2}+c\,Hash_{\mathbb{G}_1}+d\,Inv_{\mathbb{G}_1}$, where $Exp_{\mathbb{G}_1}$ denotes one exponentiation in $\mathbb{G}_1$, $Exp_{\mathbb{G}_1}$ denotes one exponentiation in $\mathbb{G}_1$, $Mul_{\mathbb{G}_1}$ denotes one multiplication in $\mathbb{G}_1$, $Pair$ denotes one pairing operation on e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, $Hash_{\mathbb{G}_1}$ denotes one hashing operation in $\mathbb{G}_1$, $Inv_{\mathbb{G}_1}$ denotes multiplicative inversion in $\mathbb{G}_1$. In fact, compared with Panda, the extra computation overhead of this new proposal is only $d\,Inv_{\mathbb{G}_1}$.
- Performance of Auditing. Pairing Based Cryptography Library is used to implement cryptographic operations. As the same as Panda, experiments are tested under Ubuntu with 2.5 GHz Processor and 4 GB Memory. Assuming the size of an element of $\mathbb{G}_1$ is $|p|=160$ bits, $|id|=80$ bits and $|n|=1{,}000{,}000$. By utilizing proper aggregation methods

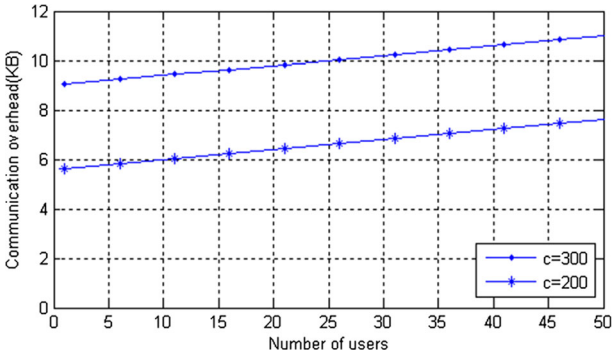


Fig. 4 Communication overhead

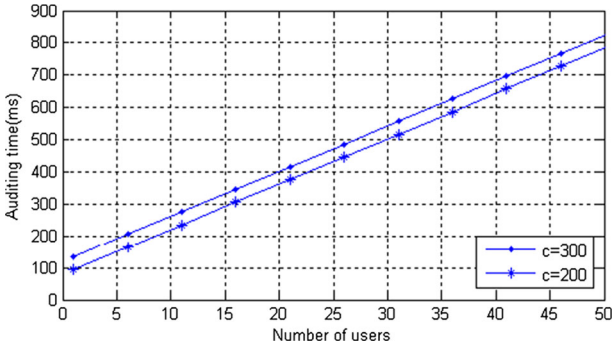


Fig. 5 Auditing time overhead

[25], the size of each block can be set as 2 KB, and the volume of shared data is 2 GB. The communication overhead and auditing time overhead are both linearly increasing with the number of users in the group, as illustrated in Figs. 4 and 5. Our scheme can also support large groups efficiently. For example, when the number of users is 50 and $c=300$, the auditing task can be finished with only 820 milliseconds and 11 KB communication overhead.

7 Conclusion

Ensuring the security of outsourced data needs continuous integrity auditing, meanwhile, without privacy leakage. In a public auditing scheme, the public verifier is delegated to check the validity of outsourced data. However, this delegation brings privacy concerns since the public verifier has the potential to derive multimedia data blocks. Moreover, if the scheme is not well designed, the cloud and internet of things might successfully hide some data corruptions for their own benefits. Besides, an outside attacker may forge an auditing proof against any auditing challenge after eavesdropping on enough valid pairs of auditing challenges and auditing proofs. In this paper, we have shown two security drawbacks of Panda. We have demonstrated that Panda is vulnerable to attacks from outside attackers and malicious cloud servers. Therefore, Panda cannot preserve data privacy or audit the integrity of shared data in the cloud and internet of things correctly. Then, we propose a new proposal for Panda, which is also a homomorphic authenticable proxy re-signature scheme. Detailed security and performance analyses show that this new proposal can provide desired data privacy preserving and sound public auditing while incurring a little extra communication and computation overhead compared with Panda. The proposed research result is able to be applied in some related research fields, such as image processing [7, 12, 13, 17, 19, 24, 43], visualization [22, 26], network [10, 11, 14, 23], grid [5, 6, 33], cloud computing [16, 36, 37, 41, 45], multimedia [9, 18, 20, 21], hardware [40], and others [8, 44].

Acknowledgments This work was supported by the school innovation foundation and the doctorial foundation under grant 2014JY170. We thank the anonymous reviewers for useful comments and suggestions.

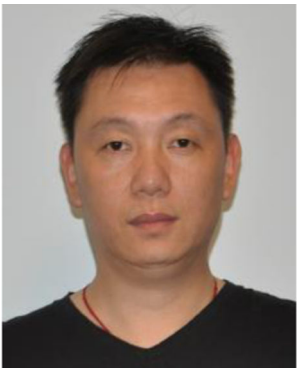
References

1. Armbrust M, Fox A, Griffith R, Joseph A, Katz R, Konwinski A et al (2010) A view of cloud computing. Commun ACM 53(4):50–8
2. Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, Song D (2007) Provable data possession at untrusted stores. In Proc CCS'07, Alexandria, VA 598–609
3. Blaze M, Bleumer G, Strauss M (1998) Divertible protocols and atomic proxy cryptography. Proc EUROCRYPT'98, Springer-Verlag 127–44
4. Boneh D, Lynn B, Shacham H (2004) Short signatures from the Weil pairing. J Cryptol 17(4):297–319
5. Che L, Shahidehpour M, Alabdulwahab A, Al-Turki Y (2015) Hierarchical coordination of a community microgrid with AC and DC microgrids. IEEE Trans Smart Grid
6. Che L, Zhang X, Shahidehpour M, Alabdulwahab A, Abusorrah A (2015) Optimal interconnection planning of community microgrids with renewable energy sources. IEEE Trans Smart Grid
7. Chen Z, Huang W, Lv Z (2016) Towards a face recognition method based on uncorrelated discriminant sparse preserving projection. Multimed Tools Appl
8. Dang S, Kakimzhanov R, Zhang M et al (2014) Smart grid-oriented graphical user interface design and data processing algorithm proposal based on LabVIEW. Environ Electr Eng (EEEIC) 14th Int Conf IEEE 323–327
9. Gu W, Lv Z, Hao M (2016) Change detection method for remote sensing images based on an improved Markov random field. Multimed Tools Appl
10. Jiang D, Xu Z, Chen Z et al (2011) Joint time–frequency sparse estimation of large-scale network traffic. Comput Netw 55(15):3533–3547
11. Jiang D, Xu Z, Li W, Yao C, Lv Z, Li T (2015) An energy-efficient multicast algorithm with maximum network throughput in multi-hop wireless networks. J Commun Netw
12. Jiang D, Xu Z, Xu H et al (2011) An approximation method of origin–destination flow traffic from link load counts. Comput Electr Eng 37(6):1106–1121
13. Jiang D, Xu Z, Zhang P, Zhu T (2014) A transform domain-based anomaly detection approach to network-wide traffic. J Netw Comput Appl 40:292–306
14. Jiang D, Ying X, Han Y et al (2015) Collaborative multi-hop routing in cognitive wireless networks. Wirel Pers Commun 1–23
15. Juels A and Kaliski BS (2007) Pors: proofs of retrievability for large files. In Proc CCS'07, Alexandria, VA 584–97
16. Li X, Lv Z, Hu J, et al (2015) Traffic management and forecasting system based on 3D GIS. 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE
17. Lin Y, Yang J, Lv Z et al (2015) A self-assessment stereo capture model applicable to the internet of things. Sensors 15(8):20925–20944
18. S Liu, W Fu, L He et al (2015) Distribution of primary additional errors in fractal encoding method [J]. Multimed Tools Appl
19. S Liu, Z Zhang, L Qi et al (2015) A fractal image encoding method based on statistical loss used in agricultural image compression [J]. Multimed Tools Appl
20. Lv Z, Halawani A, Fen S et al (2015) Touch-less interactive augmented reality game on vision based wearable device. Pers Ubiquit Comput
21. Lv Z, Halawani A, Feng S et al (2014) Multimodal hand and foot gesture interaction for handheld devices. ACM Trans Multimed Comput Commun Appl (TOMM) 11(1s):10
22. Lv Z, Tek A, Da Silva F et al (2013) Game on, science-how video game technology may help biologists tackle visualization challenges. PLoS One 8(3):57990
23. Lv Z, Yin T, Han Y, Chen Y et al (2011) WebVR—web virtual reality engine based on P2P network. J Netw 6(7):990–998
24. Ou W, Lv Z, Xie Z (2015) Spatially regularized latent topic model for simultaneous object discovery and segmentation. The 2015 I.E. International Conference on Systems, Man, and Cybernetics (SMC2015). IEEE
25. Shacham H, Waters B (2008) Compact proofs of retrievability. Proc ASIACRYPT'08 Springer-Verlag 90–107
26. Su T, Wang W, Lv Z et al (2016) Rapid Delaunay triangulation for randomly distributed point cloud data using adaptive Hilbert curve. Comput Graph 54:65–74
27. Tate SR, Vishwanathan R, Everhart L (2013) Multi-user dynamic proofs of data possession using trusted hardware. Proc ACM CODASPY 13:353–64
28. Wang B, Chow SS, Li M, Li H (2013) Storing shared data on the cloud via security-mediator. Proc IEEE ICDCS 13:124–33
29. Wang B, Li B, Li H (2012) Oruta: privacy-preserving public auditing for shared data in the cloud. Proc IEEE Cloud 12:295–302

30. Wang B, Li H, Li M (2013) Privacy-preserving public auditing for shared cloud data supporting group dynamics. Proc IEEE ICC'13, Budapest, Hungary 1946–50
31. Wang B, Li B, Li H (2013) Public auditing for shared data with efficient user revocation in the cloud. Proc IEEE INFOCOM 13:2904–12
32. Wang B, Li B, Li H (2015) Panda: public auditing for shared data with efficient user revocation in the cloud. IEEE Trans Serv Comput 8(1):92–106
33. Wang Y, Su Y, Agrawal G (2015) A novel approach for approximate aggregations over arrays. Proceedings of the 27th International Conference on Scientific and Statistical Database Management. ACM 4
34. Wang Q, Wang C, Ren K, Lou W, Li J (2011) Enabling public auditability and data dynamics for storage security in cloud computing. IEEE Trans Parallel Distrib Syst 22(5):847–59
35. Wang C, Wang Q, Ren K, Lou W (2010) Privacy-preserving public auditing for data storage security in cloud computing. Proc IEEE INFOCOM 10:525–33
36. Wang K et al (2015) Load-balanced and locality-aware scheduling for data-intensive workloads at extreme scales. Concurrency and Computation: Practice and Experience
37. Wang K et al (2015) Overcoming Hadoop scaling limitations through distributed task execution. Proc IEEE Int Conf Clust Comput
38. Worku SG, Xu C, Zhao J, He X (2013) Secure and efficient privacy-preserving public auditing scheme for cloud storage. Comput Electr Eng. doi:10.1016/j.compeleceng.2013.10.004
39. Xu C, He X, Abraha-Weldemariam D (2012) Cryptanalysis of wang's auditing protocol for data storage security in cloud computing. Proc. ICICA'12, Springer-Verlag 422–28
40. Yang J, Chen B, Zhou J et al (2015) A low-power and portable biomedical device for respiratory monitoring with a stable power source. Sensors 15(8):19618–19632
41. Yang J, He S, Lin Y, Lv Z (2016) Multimedia cloud transmission and storage system based on internet of things. Multimed Tools Appl
42. Yang K, Jia X (2013) An efficient and secure dynamic auditing protocol for data storage in cloud computing. IEEE Trans Parallel Distrib Syst 24(9):1717–26
43. Zhang S, Jing H (2014) Fast log-Gabor-based nonlocal means image denoising methods. IEEE Int Conf Image Proc (ICIP) 2014:2724–2728
44. Zhang X, Xu Z, Henriquez C et al (2013) Spike-based indirect training of a spiking neural network-controlled virtual insect. IEEE 52nd Annu Conf Decis Control (CDC) 2013:6798–6805
45. Zhang S, Zhang X, Ou X (2014) After we knew it: empirical study and modeling of cost-effectiveness of exploiting prevalent known vulnerabilities across iaas cloud. Proc 9th ACM Symp Inf Comput Commun Sec. ACM 317–328

Tonghao Yang was born in Shandong, P. R. China, in October 1987. He received his M.S. degree in information security from Zhengzhou Institute of Information Science and Technology, Zhengzhou, P. R. China, in 2012, where he is currently pursuing the Ph.D. degree in information security. His current research interests include cloud computing and cyberspace security.

Bin Yu received his B.S. degree in Dept. of Electronic Engineering from the University of Shanghai Jiaotong in 1986, the M.S. degree in Dept. of Automatic Engineering from South China University of Technology in 1991 and the Ph.D. degree in 1999. From 1997 to 1999, he worked as a research assistant at Hong Kong University of Science and Technology. From 2002 to 2003, he worked as a visiting associate professor at University of Waterloo, ON, Canada. Currently, he is a professor with Zhengzhou Institute of Information Science and Technology, Zhengzhou, P. R. China. His research interests include visual cryptography and network security.



Hengjun Wang was born in Hunan, P. R. China, in November 1973. He received his M.S. degree and Ph.D. degree from Zhengzhou Institute of Information Science and Technology, Zhengzhou, P. R. China, in 2003 and 2010, respectively. Currently, he is an associate professor with Zhengzhou Institute of Information Science and Technology. His current research interests include cloud computing and information security.

Junquan Li was born in Hebei, P. R. China, in 1965. He received his Ph.D. degree in applied mathematics from Chinese Academy of Sciences in 1990. Currently, he is a researcher with Zhengzhou Institute of Information Science and Technology, Zhengzhou, P. R. China. His current research interests include cryptography and cyberspace security.

Zhihan Lv is a native Chinese. He is an engineer and researcher of virtual/augmented reality and multimedia major in mathematics and computer science, having plenty of work experience on virtual reality and augmented reality projects, engage in application of computer visualization and computer vision. His research application fields widely range from everyday life to traditional research fields (i.e., geography, biology, medicine). During the past years, he has completed several projects successfully on PC, Website, Smartphone and Smartglasses.