SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2019, Vol. 62 032104:1–032104:16
https://doi.org/10.1007/s11432-018-9462-0

Identity-based public auditing for cloud storage systems against malicious auditors via blockchain

Jingting XUE[1*], Chunxiang XU[1*], Jining ZHAO[1,2] & Jianfeng MA[3]

[1]*Center for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu* 611731*, China;*
[2]*Department of Mathematics and Computer Science, Emory University, Atlanta* 30322*, USA;*
[3]*School of Computer Science, Xidian University, Xi'an* 710071*, China*

Received 26 February 2018/Accepted 22 May 2018/Published online 24 January 2019

Abstract Cloud storage systems provide users with convenient data storage services, which allow users to access and update outsourced data remotely. However, these cloud storage services do not guarantee the integrity of the data that users store in the cloud. Thus, public auditing is necessary, in which a third-party auditor (TPA) is delegated to audit the integrity of the outsourced data. This system allows users to enjoy on-demand cloud storage services without the burden of continually auditing their data integrity. However, certain TPAs might deviate from the public auditing protocol and/or collude with the cloud servers. In this article, we propose an identity-based public auditing (IBPA) scheme for cloud storage systems. In IBPA, the nonces in a blockchain are employed to construct unpredictable and easily verified challenge messages, thereby preventing the forging of auditing results by malicious TPAs to deceive users. Users need only to verify the TPAs' auditing results in batches to ensure the integrity of their data that are stored in the cloud. A detailed security analysis shows that IBPA can preserve data integrity against various attacks. In addition, a comprehensive performance evaluation demonstrates that IBPA is feasible and efficient.

Keywords cloud storage, public integrity auditing, identity-based cryptography, blockchain, security analysis

Citation Xue J T, Xu C X, Zhao J N, et al. Identity-based public auditing for cloud storage systems against malicious auditors via blockchain. Sci China Inf Sci, 2019, 62(3): 032104, https://doi.org/10.1007/s11432-018-9462-0

1 Introduction

As a result of the rapid development of communication technology and networks, people are now receiving, processing and storing large amounts of data daily. To mitigate the burden of local data storage, subsequent updates and maintenance, personal data are often stored on public cloud servers [1, 2]. Due to the complexity of the cloud environment, cloud servers are vulnerable to external rival attacks and internal hardware or software failures, which can cause users' data to be corrupted or even lost [3]. In addition, a cloud server is an independent and untrusted administrative entity that may delete data that users have never accessed to save storage space or hide data loss events to maintain its reputation [4]. Unfortunately, after they have uploaded their data to a cloud server, users often delete locally stored backup data. Because of these three factors, it is important for users to audit the integrity of their outsourced data on a regular basis.

To verify the integrity of outsourced data, an integrity auditing model has been proposed for periodically auditing these data [1]. Because the amount of data stored in the cloud is large and a user's

* Corresponding author (email: JTXXue@yeah.net, chxxu@uestc.edu.cn)

communication resources are limited [5], it is expensive and unreasonable for users to frequently download all of their data to audit their integrity [6]. Therefore, users traditionally authorize a third-party auditor (TPA) to conduct public audits of their outsourced data [2]. In other words, the TPA performs data integrity auditing in collaboration with the cloud server on behalf of the user.

Most public auditing schemes are based on a public key infrastructure (PKI), in which the auditor verifies users' certificates and selects the correct public key for authentication. These schemes face various issues related to certificate management, including the revocation, storage, distribution, and validation of certificates. In practice, certificate management systems are inefficient and cumbersome [7]. Furthermore, the credibility of the TPA is questionable. However, in most schemes, the auditor is assumed to be honest and reliable, which is a very strong assumption because it is quite possible for auditors to be corrupted. For example, an irresponsible auditor can establish a good integrity record without performing the public auditing protocol to reduce verification costs [8]. In this case, the aforesaid public auditing protocol do not provide any guarantee of data integrity. Consequently, it is necessary and practical to develop a public auditing scheme that can thwart malicious auditors.

Related work. To ensure the integrity of data stored on an untrusted cloud server, Juels et al. [9] proposed a "proofs of retrievability" (POR) scheme, which relies on indistinguishable blocks hidden among file blocks that serve as sentinels to detect data corruption. However, this proposal supports only a bounded number of POR queries. In accordance with the auditor's role, integrity auditing approaches can be classified into two categories: private auditing and public auditing. The POR scheme does not consider the public auditing model.

In 2007, Ateniese et al. [4] first designed a public verification scheme based on proofs of data possession (PDP), which is a proposed variant of the POR scheme that supports an unbounded number of challenge queries. Nevertheless, Ateniese et al. did not present any proof of security against arbitrary adversaries. In 2008, Shacham and Waters [10] proposed private-key-based and public-key-based POR schemes, which utilize homomorphic authenticators to yield compact proofs and support public verification.

Following the work of Shacham et al., many public auditing schemes were proposed, such as those presented in [11–23]. Considering the large data volumes involved and the limited nature of communication resources, users usually use public auditing schemes to audit the integrity of their outsourced data. These existing auditing protocols are mainly based on PKI systems. However, in a PKI-based auditing system, the problem of key management arises. To eliminate the need for certificate management in public auditing schemes, identity-based public auditing (IBPA) schemes [7, 24, 25] have been proposed.

Investigations of the credibility of the TPA have also attracted attention in the recent literature, with the intent of effectively preventing malicious auditors from forging auditing results. Many public auditing schemes depend upon a fully trusted TPA to verify the integrity of data on behalf of users without downloading the entire data set. However, a malicious TPA can reduce the number of audits to reduce resource consumption or can collaborate with cloud servers to gain certain benefits. In 2014, Armknecht et al. [8] proposed the first scheme for verifying data integrity against malicious auditors, in which the Bitcoin proof-of-work mechanism is relied upon as a secure source of time-dependent pseudorandomness. Following the work of Armknecht et al., Zhang et al. [26] proposed a certificateless public integrity verification scheme that simultaneously supports certificateless public verification and resistance against malicious auditors for verifying the integrity of outsourced data in cyber-physical-social systems. A more detailed survey on data auditing can be found in [27].

Our contributions. In this paper, we propose an identity-based public auditing scheme for ensuring data integrity in cloud storage systems, which is called IBPA. To ensure objective auditing of the integrity of outsourced data, IBPA requires the user to batch check the auditing results provided by the TPA to further confirm the integrity of the data. In IBPA, the public blockchain mechanism of the Bitcoin system is used as the key technology for public auditing. We select challenge messages for auditing data integrity based on nonces, which are indispensable features of a public blockchain that are used to solve given Hash puzzles. The nonce in a block is not predefined and is easily verifiable, which ensures that even if a malicious auditor forges an auditing result, it cannot be validated by the user. In addition, in IBPA, the TPA's auditing results are written into the public blockchain, which can serve as undeniable

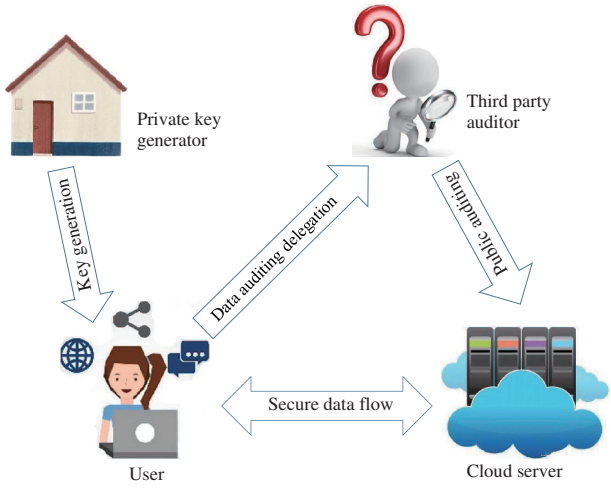

Figure 1 (Color online) System model.

evidence that the TPA has executed the auditing agreement in compliance with user requirements. Since the blockchain is inherently verifiable and resistant to modification, recording the auditing results in the blockchain ensures the traceability of the TPA's auditing services.

The main contributions of this work are summarized as follows.

(1) We propose an efficient scheme for cloud storage systems, namely, IBPA, in which challenge messages are selected based on the nonces of the public blockchain of Bitcoin. In addition, we show that the auditing results, which are written into the public blockchain, are traceable and auditable.

(2) We show that IBPA is secure under the random oracle model, and that this security is based on the computational Diffie-Hellman problem, while maintaining good efficiency. Experimental results show that the TPA requires only 80.11 s to audit 10000 data blocks and that a user requires only 80.14 s to generate authentication tags and verify the auditing results for these data.

Organization. The remainder of this article is organized as follows. We formulate the problem in Section 2 and present preliminaries in Section 3. In Section 4, we describe the construction of the proposed IBPA scheme. Then, we prove the correctness and security of IBPA in Section 5. In Section 6, we present a performance evaluation. Finally, we summarize our conclusion and discuss future research directions in Section 7.

2 Problem formulation

2.1 System model

The system model of IBPA, as shown in Figure 1, involves four entities: a private key generator (PKG), a user, a cloud server (CS), and a TPA.

- The PKG is governed by a fully trusted authority, which sets the system parameters and generates private keys for each user.
- The CS is managed by a cloud service provider and provides users with cloud storage services. The CS possesses an enormous amount of storage space and powerful computational capabilities. However, the cloud service provider may be a dishonest entity and may hide data corruption or loss.
- The user is an entity with large amounts of data and limited communication resources. The user uploads local data to the CS, as permitted by the (paid) cloud storage service.
- The TPA is delegated by the user to audit the integrity of the outsourced data. The TPA has the expertise and ability to complete the auditing task but may not do so in full accordance with the users' audit requirements.

Here, we briefly describe the relationships among the entities in the system model. After the user's local data have been uploaded to the CS, the user relies on the CS to store and maintain those data. The user can access and update the outsourced data anytime and anywhere. To ensure the integrity of the data, the user delegates the TPA to regularly audit the data and checks the TPA's auditing results over a longer period of time.

However, the TPA may perform fewer audits than agreed upon with the user to reduce auditing costs, or, for financial reasons, the TPA and the CS may collude to forge audit data to deceive the user.

IBPA is formally defined as follows.

Definition 1 (IBPA)**.** IBPA consists of seven algorithms: Setup, Keyextr, Store, Challen, Proofgen, Audit, and Checklog.

Setup$(k) \rightarrow$ (Para, s). Based on an input security parameter k, the Setup algorithm establishes a public parameter Para for the system and a master secret key s.

Keyextr(Para, s, ID) $\rightarrow (sP_{\mathcal{U},0}, sP_{\mathcal{U},1})$. From the input public parameter Para, the master secret key s and a user's identity ID, the Keyextr algorithm generates a private key $(sP_{\mathcal{U},0}, sP_{\mathcal{U},1})$ and a common state parameter ω for the user.

Store(Para, F, $(sP_{\mathcal{U},0}, sP_{\mathcal{U},1})) \rightarrow \phi$. From the input public parameter Para, a file F and a user's private key $(sP_{\mathcal{U},0}, sP_{\mathcal{U},1})$, the Store algorithm generates a set ϕ of authentication tags that correspond to the data blocks in file F for the user.

Challen(Para, t) $\rightarrow D$. From the input public parameter Para and a time t that is specified by the user, the Challen algorithm generates a challenge message D for the TPA.

Proofgen(Para, D) $\rightarrow C$. From the input public parameter Para and a challenge message D, the Proofgen algorithm generates proof information C for the CS.

Audit(Para, C) $\rightarrow$ 0/1. From the input public parameter Para and proof information C, the Audit algorithm outputs an auditing result of either 0 or 1 for the TPA, where 0 means reject and 1 means accept.

Checklog(Para, Lf) $\rightarrow$ 0/1. From the input public parameter Para and auditor's log file Lf, the Checklog algorithm outputs the checking result as 0 or 1 for the user, where 0 means reject and 1 means accept.

2.2 Threat model

In the threat model, we consider three types of attacks: forgery, replacement, and replay attacks. Here, we define the role and possible behaviors of each entity. The PKG is completely trustworthy and will not launch any attack. The CS is not trusted since it may hide data loss to maintain a good reputation or may delete data that the user has never accessed to save storage space. The TPA is semicredible and may deviate from the protocol to reduce auditing overhead and/or in collusion with the CS [8]. For example, a malicious auditor may share the secret key with the CS so that both can generate correct POR without having to store the outsourced data at all. We assume that the user will not attack the IBPA scheme.

- Replacement attack. An adversary attempts to pass the data integrity audit by replacing the challenged block and signature with an unchallenged and uncorrupted block and signature.
- Forgery attack. An adversary forges proof information to deceive the TPA or the user or forges an auditing result to cheat the user.
- Replay attack. An adversary replays previous proof information in an attempt to pass the TPA's audit.

We consider that the CS may launch all of the above attacks and that the TPA may launch forgery attacks. In addition, we consider that external adversaries may launch forgery and replay attacks.

2.3 Design goals

In this paper, we target public integrity auditing for cloud storage systems, in which three challenging problems are encountered.

(1) Constructing an unbiased, unpredictable and easily verified challenge message. To ensure correct and effective auditing, in order to prevent a malicious auditor from generating an audit result ahead of

time and provide incontrovertible evidence that the TPA has properly executed the auditing protocol, the challenge information must not depend solely on either the user or the auditor and cannot be predefined. Therefore, unbiased construction of the challenge information is crucial for effective audit execution.

(2) Resisting forgery attacks on tags. The user uploads the processed files and tags constituting the data to be stored to the CS. If the CS or a third party can forge an authentication tag, the CS can still pass the audit of the TPA if the original data are lost. Therefore, resistance against tag forgery is a serious consideration in the tag construction process.

(3) Verifying the auditing results effectively. To prevent the auditor from forging auditing results, the user must regularly verify the auditor's auditing results. Note that the cycle of user verification of auditing results is much longer than that of the auditor's auditing of data integrity. Thus, in cases where the user authorizes a TPA to audit data, the procedure for validating the auditing results should be more efficient than that for auditing the data stored in the cloud.

To ensure efficient integrity auditing for outsourced data on a CS under the previously described threat model, IBPA should achieve the following three sets of objectives.

Functionality:

• Public auditability. Anyone (not only the TPA) is allowed to audit the integrity of user data that are stored in the cloud. That is, anyone can easily generate an unbiased challenge message and perform data auditing with low resource consumption.

• Storage correctness. The CS must pass the TPA's data audit only if the CS is storing the user's data correctly.

• Resistance against malicious auditors. The TPA's verification results can be reviewed by the user only if (i) the challenge information is selected in accordance with the user's requirements and (ii) the user's data are stored correctly and completely in the CS.

Security:

• Blockless verification. During the auditing procedure, the TPA does not require and cannot retrieve any data blocks that the user has stored in the cloud.

• Privacy preservation. The TPA cannot obtain any real information about the user's data from the received auditing materials.

Efficiency:

• Light weight. The public audit should be performed with low communication overhead and a low computational cost. These characteristics allow IBPA to be applied on various resource-constrained terminals.

3 Preliminaries

3.1 Bilinear maps and computational Diffie-Hellman assumption

Definition 2 (Bilinear maps)**.** Let G_1 be an additive group of large prime order p, let G_2 be a multiplicative group of the same order, and let P be a generator in G_1. Let e: $G_1 \times G_1 \to G_2$ be a bilinear mapping with the following three properties:

(a) Bilinear. $\forall P, Q \in G_1$ and $\forall a, b \in Z_p$, $e(aP, bQ) = e(P, Q)^{ab}$ holds;

(b) Non-degenerate. $\forall P \in G_1$, $e(P, P) \neq 1$, where 1 is the identity element;

(c) Efficiently computable. $\forall P, Q \in G_1$, there exists an efficiently computable algorithm for computing $e(P, Q)$.

The security of our scheme is based on the assumed hardness of the computational Diffie-Hellman (CDH) problem on groups.

Definition 3 (CDH assumption)**.** Given $P, aP, bP \in G_1$, where $a, b \in Z_p$ are randomly chosen, no t-time algorithm (Algo) has an advantage ε in computing abP. That is, $|\Pr[\text{Algo}(P, aP, bP)] = abP| \leqslant \varepsilon$.

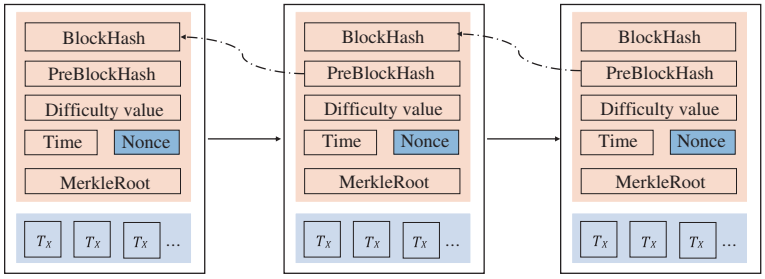

Figure 2 (Color online) Simplified blockchain.

3.2 Blockchain

Blockchain is well known for its outstanding performance in a variety of cryptocurrency systems, such as Bitcoin [28], Ethereum [29], and Litecoin[1)]. A public blockchain is an immutable ledger that is used to record transactions, which represent the states of users in the system, and users can conduct transactions without referring to a central authority. Structurally, a public blockchain is a one-way linear set of data units, in which each unit of data is called a block. Each block contains a pointer PreBlockHash that points to the previous block, a time stamp time that records the time when the block was added to the blockchain, and multiple transactions T_Xs that record approved transactions, as shown in Figure 2. All of the blocks are linked in chronological order and are protected by a cryptographic Hash function for integrity. Therefore, the blockchain is inherently verifiable, and its recorded transactions are resistant to tampering.

A blockchain system includes miners who verify the validity of each transaction and maintain the blockchain. Depending on the type of participants involved, blockchains are divided into two types: private blockchains (including consortium blockchains) and public blockchains. In a public blockchain, each participant can become a miner, and a transaction can be recorded to the blockchain only if it has been approved and accepted by most miners. Additional technical details on public blockchains can be found in [30, 31]. Before a new block is generated, miners collect as many transactions as possible and vary the nonce until one of them finds a solution to a given Hash puzzle. In the blockchain of the Bitcoin system's source code, the nonce size is defined as Int32. The complete randomness of the nonce guarantees the unpredictability of the blocks. In the IBPA scheme, a nonce is used in Challen to generate an unbiased, unpredictable and unforgeable challenge message. Furthermore, after completing the integrity auditing of user data via Audit, the TPA broadcasts the log file that contains the auditing results to the network. Once the log file has been validated and added to the public blockchain, the auditing results cannot be compromised by a misbehaving CS or a malicious auditor. Specifically, in the Audit algorithm, the TPA writes a challenge message to a log file and broadcasts the log file to the blockchain network. The immutability and openness of the blockchain endow the data stored in the blockchain with traceability. That is, the inherent properties of the blockchain guarantee the traceability of all contents of the log file, including the challenge message.

4 Proposed scheme

4.1 Overview of IBPA

We propose IBPA to enable the efficient public auditing of outsourced data in cloud storage systems, including the abilities to resist malicious auditors, construct challenge messages, and efficiently verify auditing results. To achieve these goals, the core strategy of IBPA is to (a) construct truly random challenge messages and (b) store the auditing results generated by TPAs in a public blockchain. Specifically, IBPA uses nonces in the Bitcoin system to sample challenges based on time. The different nonces in blocks associated with different times are random numbers, which are unpredictable and traceable. This design

1) The cryptocurrency for payments based on blockchain technology. http://litecoin.org.

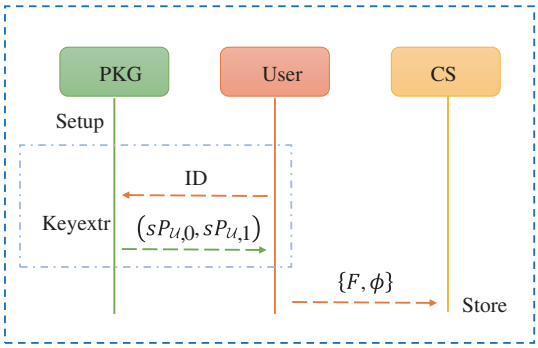

Figure 3 (Color online) Procedure for the setup phase.

deprives the auditor and the user of the ability to subjectively choose challenge messages. This system has several important properties: (i) Even if the user's data are lost, the auditor may be able to provide sufficient evidence that he has properly complied with the audit protocol; (ii) Random challenge messages prevent malicious auditors from generating forged auditing results beforehand or colluding with CSs; (iii) The system truly achieves "public auditing" because anyone can obtain a nonce in the blockchain to generate a challenge message. In addition, to improve the efficiency of performing user verification of auditing results, we define the period for the user verification of auditing results to be much longer than that for the auditing of data integrity by the TPA. In other words, we allow a user to verify multiple auditing results at the same time.

4.2 Construction of IBPA

A PKG, a user $\mathcal{U}$, a CS, and a TPA participate in both phases (the setup phase and the audit phase) of IBPA.

A file F is preprocessed into n blocks, $F = m_1 \| m_2 \| \cdots \| m_n$, where $m_j \in Z_p$, $j \in [1, n]$, and p is a large prime.

Setup phase. This phase is illustrated in Figure 3.

(1) Setup. The PKG generates the system parameters and a master secret key.

• With a security parameter k, the PKG selects two groups G_1 and G_2 of the same order p and a bilinear pairing e: $G_1 \times G_1 \to G_2$.

• Let P be a generator of group G_1. Choose a random $s \in Z_p$ and compute the public key $Q = sP$.

• Define Hash functions $H_1, H_2 : \{0,1\}^* \to G_1$, $h : G_1 \to Z_p$, and $H : \{0,1\}^* \to Z_p$.

The system parameters are $\text{Para} = \{G_1, G_2, e, H_1, H_2, h, H\}$. The PKG's master secret key is s.

(2) Keyextr. User $\mathcal{U}$ registers with the PKG to obtain a private key.

• $\mathcal{U}$ submits his identity ID to the PKG.

• The PKG computes

$$P_{\mathcal{U},0} = H_1(\text{ID}, 0), \tag{1}$$

$$P_{\mathcal{U},1} = H_1(\text{ID}, 1). \tag{2}$$

• $\mathcal{U}$ receives the private key $(sP_{\mathcal{U},0}, sP_{\mathcal{U},1})$ and a common state parameter ω.

(3) Store. $\mathcal{U}$ signs block m_j and stores the data and the authentication tag set in the cloud.

• Choose a random $r \in Z_p$ and a random element name for file naming and compute

$$(S_j, T_j) = (rH_2(\omega||j) + H(\text{name}||j)sP_{\mathcal{U},0} + m_j sP_{\mathcal{U},1}, rP) \tag{3}$$

as the authentication tag.

• Denote the set of authentication tags by

$$\phi = \{(S_j, T_j)\}_{j \in [1,n]}. \tag{4}$$

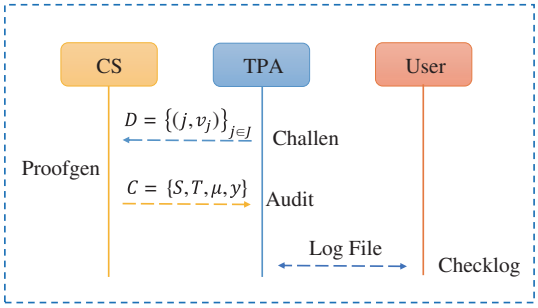

Figure 4 (Color online) Procedure for the audit phase.

Table 1 Log file

t	Nonce	D	(S, T, μ, y)	Auditing results
t_1	nonce_1	D_1	(S_1, T_1, μ_1, y_1)	1/0
t_2	nonce_2	D_2	(S_2, T_2, μ_2, y_2)	1/0
		$\vdots$		

• Send $\{F, \phi\}$ to the CS and delete these data from the local storage.

Audit phase. This phase is illustrated in Figure 4.

(1) Challen. The TPA generates a challenge message.

• Obtain the nonce in the corresponding block based on the time t that is specified by $\mathcal{U}$.

• Choose a random l-element subset $J = \{a_1, a_2, \ldots, a_l\}$ of the set $[1, n]$ based on the nonce and k.

• Choose a random $v_j \in Z_p$ for each $j \in J$.

• Generate a challenge message:

$$D = \{j, v_j\}_{j \in J}, \tag{5}$$

and send it to the CS.

(2) Proofgen. The CS generates proof information.

• Receive $D = \{j, v_j\}_{j \in J}$ and choose a random number $x \in Z_p$.

• Compute

$$\mu = x^{-1} \left(\sum_{j=a_1}^{a_l} m_j v_j + h(y) \right) \in Z_p, \tag{6}$$

$$y = x P_{\mathcal{U},1} \in G_1, \tag{7}$$

$$(S, T) = \left(\sum_{j=a_1}^{a_l} v_j S_j, \sum_{j=a_1}^{a_l} v_j T_j \right). \tag{8}$$

• Send the proof information:

$$C = \{S, T, \mu, y\} \tag{9}$$

to the TPA.

(3) Audit. The TPA audits the integrity of the challenged block.

• Verify

$$e(S, P) \stackrel{?}{=} e \left(\sum_{j=a_1}^{a_l} H_2(\omega || j) v_j, rP \right) \cdot e \left(\sum_{j=a_1}^{a_l} H(\text{name} || j) v_j P_{\mathcal{U},0} + \mu y - h(y) P_{\mathcal{U},1}, Q \right), \tag{10}$$

and output 1 to indicate acceptance if the equation holds; otherwise, output 0 to indicate rejection.

• Create an entry $(t, \text{nonce}, D, (S, T, \mu, y), 0/1)$ and store all such entries (one audit) in a log file in chronological order as shown in Table 1.

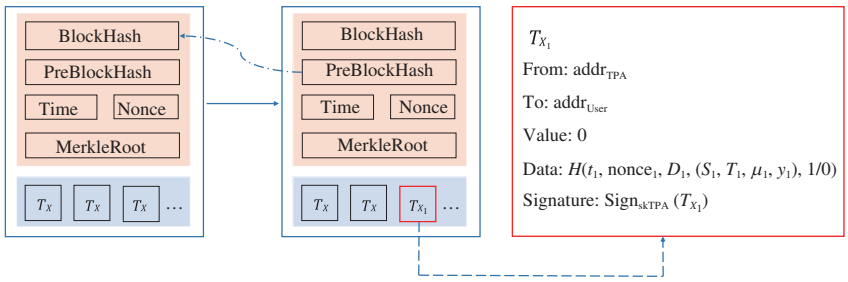

Figure 5 (Color online) Public blockchain.

• Computer the Hash value (take the first auditing task for example):

$$A_1 = H(t_1, \text{nonce}_1, D_1, (S_1, T_1, \mu_1, y_1), 1/0). \tag{11}$$

• Generate a transaction T_{X_1} as shown in Figure 5, where the data is a set of A_1, and upload it to the blockchain.

(4) Checklog. $\mathcal{U}$ checks the validity of the log file that is recorded in the public blockchain.

• Check whether $D_1, D_2, \ldots$ were chosen in accordance with the nonce in the corresponding block at the specified time t.

• Pick a random subset B of the challenge message and generate a set $B = \{b_1, b_2, \ldots, b_{l'}\}$.

• Check $\{S^{(B)}, T, \mu, y\}$ in the log file, where $S^{(B)} = \sum_{j=b_1}^{b_{l'}} v_j S_j$.

• Verify

$$e(S^{(B)}, P) \stackrel{?}{=} e\left(\sum_{j=b_1}^{b_{l'}} H_2(\omega||j)v_j, rP\right) \cdot e\left(\sum_{j=b_1}^{b_{l'}} H(\text{name}||j)v_j P_{\mathcal{U},0} + \mu y - h(y)P_{\mathcal{U},1}, Q\right). \tag{12}$$

If the verification fails, $\mathcal{U}$ considers the cloud-stored data to be corrupted and either/both the TPA or/and the CS to be malicious. When this occurs, $\mathcal{U}$ outputs a verification result of 0 to indicate rejection. Otherwise, an output of 1 is generated to indicate acceptance.

4.3 Remark

To resist malicious TPAs, a straightforward solution is to require the TPA to send the log file to $\mathcal{U}$ after each auditing task is performed. In the case, $\mathcal{U}$ has to keep online to receive the log file and thereby incurs heavy communication overhead. To release $\mathcal{U}$ from these overhead, the TPA may utilize a store-and-forward system (e.g., cloud-based email system) to send the log file, which enables $\mathcal{U}$ to access the log file when the Internet is available. Nevertheless, this essentially introduces a trusted entity to resist malicious auditors, since a service provider who provides the cloud-based email service must be honest and reliable. Therefore, the TPA is required to upload the log file to the blockchain rather than directly to $\mathcal{U}$ in the Audit algorithm.

5 Security proof for IBPA

5.1 Correctness proof

The correctness of (10) is derived as follows:

$$\begin{aligned} e(S, P) &= e\left(\sum_{j=a_1}^{a_l} v_j S_j, P\right) \\ &= e\left(\sum_{j=a_1}^{a_l} v_j(rH_2(\omega||j) + H(\text{name}||j)sP_{\mathcal{U},0} + m_j sP_{\mathcal{U},1}), P\right) \end{aligned}$$

$$= e\left(\sum_{j=a_1}^{a_l} v_j r H_2(\omega||j), P\right) \cdot e\left(\sum_{j=a_1}^{a_l} v_j H(\text{name}||j) s P_{\mathcal{U},0} + \sum_{j=a_1}^{a_l} v_j m_j s P_{\mathcal{U},1}, P\right)$$
$$= e\left(\sum_{j=a_1}^{a_l} H_2(\omega||j) v_j, rP\right) \cdot e\left(\sum_{j=a_1}^{a_l} H(\text{name}||j) v_j P_{\mathcal{U},0} + \mu y - h(y) P_{\mathcal{U},1}, Q\right).$$

The correctness of (11) is derived as follows:

$$\begin{aligned} e(S^{(B)}, P) &= e\left(\sum_{j=b_1}^{b_{l'}} v_j S_j, P\right) \\ &= e\left(\sum_{j=b_1}^{b_{l'}} v_j (r H_2(\omega||j) + H(\text{name}||j) s P_{\mathcal{U},0} + m_j s P_{\mathcal{U},1}), P\right) \\ &= e\left(\sum_{j=b_1}^{b_{l'}} v_j r H_2(\omega||j), P\right) \cdot e\left(\sum_{j=b_1}^{b_{l'}} v_j H(\text{name}||j) s P_{\mathcal{U},0} + \sum_{j=b_1}^{b_{l'}} v_j m_j s P_{\mathcal{U},1}, P\right) \\ &= e\left(\sum_{j=b_1}^{b_{l'}} H_2(\omega||j) v_j, rP\right) \cdot e\left(\sum_{j=b_1}^{b_{l'}} H(\text{name}||j) v_j P_{\mathcal{U},0} + \mu y - h(y) P_{\mathcal{U},1}, Q\right). \end{aligned}$$

5.2 Resistance against attacks

Theorem 1. IBPA can resist replacement attacks from the CS.

Proof. Suppose that block m_k of file F has been deleted but that two blocks m_{k_1} and m_{k_2} and

$$\delta_{k_1} = (S_{k_1}, T_{k_1}), \quad \delta_{k_2} = (S_{k_2}, T_{k_2}) \tag{13}$$

are well maintained in the cloud, where $k, k_1, k_2 \in [1, n]$. During the auditing process, both the TPA and the user honestly execute the scheme. That is, the user computes

$$S_j = r H_2(\omega||j) + H(\text{name}||j) s P_{\mathcal{U},0} + m_j s P_{\mathcal{U},1}, \quad T_j = rP \tag{14}$$

in the Store phase. The CS sends the proof information $C = \{S, T, \mu, y\}$ to the TPA in the Proofgen phase. Since

$$(S_{k_1}, T_{k_1}) = (r H_2(\omega||k_1) + H(\text{name}||k_1) s P_{\mathcal{U},0} + m_{k_1} s P_{\mathcal{U},1}, rP), \tag{15}$$
$$(S_{k_2}, T_{k_2}) = (r H_2(\omega||k_2) + H(\text{name}||k_2) s P_{\mathcal{U},0} + m_{k_2} s P_{\mathcal{U},1}, rP), \tag{16}$$

it follows that

$$\begin{aligned} S_k^* &= \alpha_{k_1} S_{k_1} + \alpha_{k_2} S_{k_2} \\ &= \alpha_{k_1} (r H_2(\omega||k_1) + H(\text{name}||k_1) s P_{\mathcal{U},0} + m_{k_1} s P_{\mathcal{U},1}) \\ &\quad + \alpha_{k_2} (r H_2(\omega||k_2) + H(\text{name}||k_2) s P_{\mathcal{U},0} + m_{k_2} s P_{\mathcal{U},1}) \\ &= (\alpha_{k_1} H_2(\omega||k_1) + \alpha_{k_2} H_2(\omega||k_2)) r + (\alpha_{k_1} H(\text{name}||k_1) \\ &\quad + \alpha_{k_2} H(\text{name}||k_2)) s P_{\mathcal{U},0} + (\alpha_{k_1} m_{k_1} + \alpha_{k_2} m_{k_2}) s P_{\mathcal{U},1} \\ &\neq H_2(\omega||k) r + H(\text{name}||k) s P_{\mathcal{U},0} + m_k^* s P_{\mathcal{U},1}, \end{aligned} \tag{17}$$
$$T_k^* = \alpha_{k_1} T_{k_1} + \alpha_{k_2} T_{k_2} = (\alpha_{k_1} + \alpha_{k_2}) rP. \tag{18}$$

The probability that the following three equations are satisfied simultaneously is negligible:

$$m_{k_1} \cdot \alpha_{k_1} + m_{k_2} \cdot \alpha_{k_2} = m_k^*, \tag{19}$$
$$H(k_1) \cdot \alpha_{k_1} + H(k_2) \cdot \alpha_{k_2} = H(k), \tag{20}$$

$$H(\omega\|k_1)\cdot\alpha_{k_1}+H(\omega\|k_2)\cdot\alpha_{k_2}=H(\omega\|k). \tag{21}$$

That is, $\delta_k^*=\{S_k^*,T_k^*\}$ cannot pass the audit of the TPA. Therefore, our scheme can resist replacement attacks.

Theorem 2. IBPA can resist forgery attacks from the CS or the TPA.

Proof. Suppose that there exists an adversary who modifies data block m_k to $m_k^*=m_k+l_k$ for $k\in[1,n]$. During the auditing process, both the TPA and the CS honestly execute the scheme. That is, in the Store phase, the TPA sends the challenge message $D=\{(j,v_j)\}_{j\in J}$ to the CS. In the Proofgen phase, the CS computes the following:

$$\mu^*=\sum_{k=1}^{n}(m_k+l_k)\cdot v_k, \tag{22}$$

$$\begin{aligned}\widehat{\mu}&=x^{-1}\cdot(\mu^*+h(y))\\&=x^{-1}\cdot\left(\sum_{k=1}^{n}m_kv_k+\sum_{k=1}^{n}l_kv_k+h(y)\right)\\&=x^{-1}\cdot\sum_{k=1}^{n}m_kv_k+x^{-1}\cdot\sum_{k=1}^{n}l_kv_k+x^{-1}\cdot h(y)\\&=\mu+x^{-1}\cdot\sum_{k=1}^{n}l_kv_k.\end{aligned} \tag{23}$$

Then, the CS sends the proof information $C=\{S,T,\widehat{\mu},y\}$ to the TPA. The adversary intercepts C on the communication channel. However, for the adversary to modify the original proof information to valid proof information, he must modify $\widehat{\mu}$ to μ. That is, he must compute $\widehat{\mu}-x^{-1}\cdot\sum_{k=1}^{n}l_kv_k$. Note that x is randomly chosen by the CS and that v_k is randomly chosen by the TPA. Normally, x and v_k cannot both be known by the same adversary; therefore, the adversary cannot pass the verification. Hence, our scheme can resist forgery attacks.

Theorem 3. IBPA can resist replay attacks from the CS.

Proof. If the CS has deleted or lost m_k, it may attempt to execute a replay attack to pass the audit by using another block m_i and its data authentication tag (S_i,T_i). Then, the CS calculates the proof information (S^*,T^*) as follows:

$$S^*=v_jS_i+\sum_{j\in J,j\neq k}v_jS_j,\quad T^*=v_jT_i+\sum_{j\in J,j\neq k}v_jT_j. \tag{24}$$

We have the following verification process:

$$\begin{aligned}e(S^*,P)&=e\left(v_jS_i+\sum_{j\in J,j\neq k}v_jS_j,P\right)\\&=e(v_j(rH_2(\omega||i)+H(\text{name}||i)sP_{\mathcal{U},0}+m_jsP_{\mathcal{U},1}),P)\\&\quad\cdot e\left(\sum_{j\in J,j\neq k}v_j(rH_2(\omega||j)+H(\text{name}||j)sP_{\mathcal{U},0}+m_jsP_{\mathcal{U},1}),P\right)\\&=e(v_jH_2(\omega||i),rP)+e(v_jH(\text{name}||i)P_{\mathcal{U},0}+v_jm_iP_{\mathcal{U},1},sP)\\&\quad\cdot e\left(\sum_{j\in J,j\neq k}v_jH_2(\omega||j),rP\right)+e\left(\sum_{j\in J,j\neq k}v_jH(\text{name}||j)P_{\mathcal{U},0}+\sum_{j\in J,j\neq k}v_jm_jP_{\mathcal{U},1},sP\right)\\&=e\left(v_jH_2(\omega||i)+\sum_{j\in J,j\neq k}v_jH_2(\omega||j),rP\right)\end{aligned}$$

$$\cdot e\left(\left(H(\text{name}||i)v_j+\sum_{j\in J,j\neq k}H(\text{name}||j)v_j\right)P_{\mathcal{U},0}+\left(\sum_{j\in J,j\neq k}m_jv_j+m_iv_j\right)P_{\mathcal{U},1},sP\right)$$
$$=e\left(\sum_{j=a_1}^{a_l}H(\text{name}||j)v_jP_{\mathcal{U},0}+(H(\text{name}||i)-H(\text{name}||k))v_jP_{\mathcal{U},0}\right.$$
$$\left.+\left(\sum_{j=a_1}^{a_l}m_jv_j+m_iv_j-m_kv_j\right)P_{\mathcal{U},1},sP\right)\cdot e\left(\sum_{j=a_1}^{a_l}H_2(\omega||j)v_j+H_2(\omega||i)v_j-H_2(\omega||k)v_j,rP\right).$$

If the proof information (S^*,T^*) can pass the audit of the TPA, then

$$H_2(\omega||i)v_j-H_2(\omega||k)v_j=0, \tag{25}$$
$$H(\text{name}||i)-H(\text{name}||k)=0, \tag{26}$$
$$m_iv_j-m_kv_j=0 \tag{27}$$

must hold simultaneously. Since the Hash functions $H_2(\cdot)$ and $H(\cdot)$ do not collide, we know that

$$H_2(\omega||i)v_j-H_2(\omega||k)v_j\neq 0, \tag{28}$$
$$H(\text{name}||i)-H(\text{name}||k)\neq 0. \tag{29}$$

In other words, the proof information (S^*,T^*) shows that the CS-generated data cannot pass the data integrity audit. Therefore, our scheme can resist replay attacks.

Theorem 4. If an adversary can forge valid proof information to pass an audit, then he can solve the CDH problem.

Proof. The integrated storage of the outsourced data means that no adversary in the validated game can forge, with a non-negligible probability, the correct calculations to make the auditing program output acceptable.

The adversary is well trained and can forge valid calculated values. The challenger holds a large number of validated value lists for responding to queries. The challenger and the adversary can observe all cases in the verification process. If the adversary in one case makes the validation output acceptable except for the values $(S',T')\neq(S,T)$, the challenger announces failure and aborts the game.

$$(S,T)=\left(\sum_{j=a_1}^{a_l}v_jS_j,\sum_{j=a_1}^{a_l}v_jT_j\right), \tag{30}$$

where D is the TPA's challenge message and S_j and T_j are the authentication tag values in the file blocks.

After receiving the challenge message D, if the server responds with $C=\{S,T,\mu,y\}$, the adversary responds with $C'=\{S',T',\mu',y\}$ by validating the equation. Because C' causes the challenger to abort the game, $(S',T')\neq(S,T)$. By the scheme's correctness, the response should satisfy the following auditing equations:

$$e(S',P)=e(T',H_2(\omega||j))\cdot e\left(\sum_{j=a_1}^{a_l}H(\text{name}||j)v_jP_{i,0}+\mu'y-h(y)P_{i,1},Q\right), \tag{31}$$
$$e(S,P)=e(T,H_2(\omega||j))\cdot e\left(\sum_{j=a_1}^{a_l}H(\text{name}||j)v_jP_{i,0}+\mu y-h(y)P_{i,1},Q\right). \tag{32}$$

Clearly, $\mu'\neq\mu$, followed by the verification, or $(S',T')=(S,T)$, which is contrary to the above assumption. Therefore, we define $\Delta\mu=\mu'-\mu$. Now, we will show how an adversary can construct a simulator to solve the CDH problem, namely, calculate sP' from P, sP, and P'.

During the Setup algorithm, $Q = sP$ is computed as the segmental public key for the PKG without awareness of its secret s. The simulator operates the random oracles H, H_1 and H_2 and stores lists of queries to respond coincidentally. Then, it responds to the adversary's queries of $H(j)$ as follows.

The simulator randomly selects $b, d, \beta \in Z_q$. For $\{\mathrm{ID}_i, m_j\}$, it computes

$$H(\mathrm{name}||j) = -d^{-1}bm_j, \tag{33}$$

and

$$H_2(\omega||j) = \beta P. \tag{34}$$

For random numbers $a, c, r_j \in Z_q$, it calculates

$$P_{i,1} = H_1(\mathrm{ID}_i, 1) = aP + bP', \quad P_{i,0} = H_1(\mathrm{ID}_i, 0) = cP + dP'. \tag{35}$$

Now, (S_j, T_j) can be obtained

$$\begin{aligned} &r_i H_2(\omega||j) + H(\mathrm{name}||j)sP_{i,0} + m_j sP_{i,1} \\ &= r_i\beta P + (-d^{-1}bm_j)\cdots(cP + dP') + m_j s(aP + bP') \\ &= r_i\beta P + (a - d^{-1}bc)m_j Q. \end{aligned} \tag{36}$$

Because $T_j = r_i P$, the simulator can compute the authentication tag for the data block:

$$(S_j, T_j) = (r_i\beta P + (a - d^{-1}bc)m_j Q, r_i P). \tag{37}$$

The interaction between the simulator and the adversary continues until the following situation arises: during the verification process in the protocol, the adversary successfully obtains an authentication tag (S', T') that is different from the expected verification tag (S, T). We know that the parameters associated with the protocol instance are generated by the emulator and will be used as part of the Sign algorithm. Since $H_2(\omega||j) = \beta P$, $Q = sP$, we have

$$\begin{aligned} e(S, P) &= e(T, H_2(\omega||j))e\left(\sum_{j=a_1}^{a_l} H(\mathrm{name}||j)v_j P_{i,0} + \mu y - h(y)P_{i,1}, Q\right) \\ &= e\left(\beta T + s\left(\sum_{j=a_1}^{a_l} H(\mathrm{name}||j)v_j P_{i,0} + \mu y - h(y)P_{i,1}\right), P\right). \end{aligned} \tag{38}$$

We compute $e(S' - S, P) = e(\beta(T' - T) + s\Delta\mu P_{i,1}, P)$. When rescheduling the terms with $P_{i,1} = aP + bP'$, we have $S' - S + \beta(T - T') - \Delta\mu axQ = \Delta\mu bsxP'$. Namely, there is a solution to the CDH problem:

$$sP' = \frac{1}{\Delta\mu bx}(S' - S + \beta(T - T') - \Delta\mu axQ). \tag{39}$$

It is clear that $\Delta\mu \neq 0$. The random number b is information that was hidden from the adversary. The denominator is 0 only with a negligible probability of $1/q$. Thus, we can construct a simulator that the adversary can employ to solve the CDH problem with a high probability of $1 - 1/q$.

6 Performance evaluation

In this section, we evaluate the security of the IBPA scheme and compare the computational overhead and communication cost of our scheme with those of the existing IBRDIC [32] and FIBDIA [33] schemes. All experiments were conducted with type A pairing using the PBC library on a Chinese-made HP desktop computer equipped with an Intel Core i5 CPU and 4 GB of RAM. In the implementation, we chose parameters that represent an 80-bit security level: $|q| = 512$ bits and $|p| = 160$ bits. Notations for

Table 2 Notations for operations/implications

Symbol	Corresponding operation/implication
M	The point multiplication operation in G_1
E	The exponentiation operation in G_2
P	The pairing operation
$\|x\|$	The number of bits of x

Table 3 Comparison of costs

Scheme	User's computational cost	TPA's computational cost	TPA's communication cost
IBRDIC [32]	$(n+2)E$	$(n+3)E+(n+1)P$	$\|m\|+2\|G_1\|$
FIBDIA [33]	$nM+4nE$	$(4n+1)E+(n+2)P$	$\|m\|+3\|G\|$
Ours	$(n+4)M+3P$	$(n+1)M+3P$	$\|Z_q\|+3\|G_1\|$

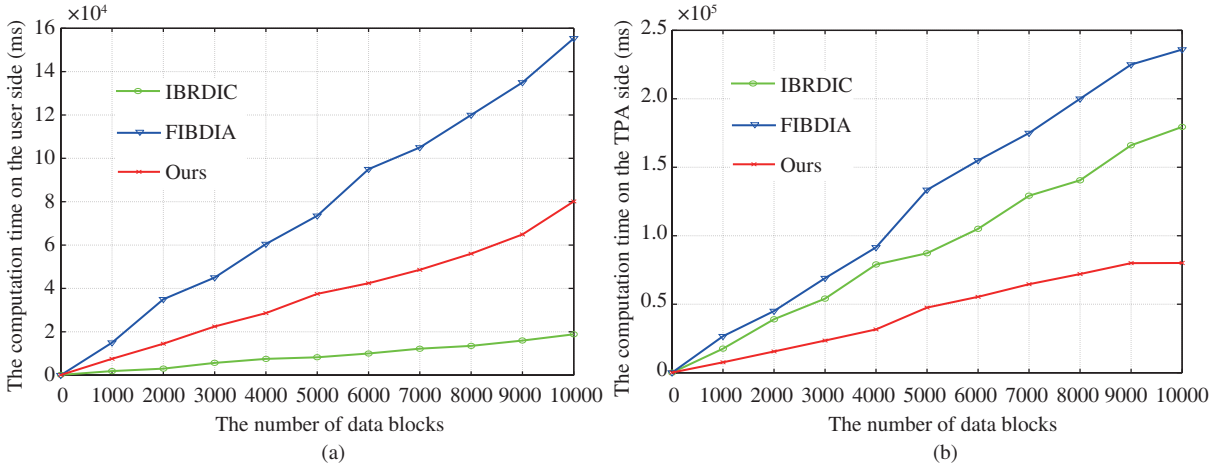

Figure 6 (Color online) (a) Computation time on the user side versus the number of data blocks; (b) computation time on the TPA side versus the number of data blocks.

some operations and implications are specified in Table 2; other operations are ignored since listing the operations requires the longest running time of the whole procedure.

According to Table 3, our scheme has a lower computational overhead than that of either IBRDIC or FIBDIA on both the user side and the TPA side. Specifically, for the user-side computational cost, our scheme requires only $(n+4)M+3P$, which is less than IBRDIC's $(n+2)E$ and FIBDIA's $nM+4nE$. The computational cost on the TPA side of our scheme is $(n+1)M+3P$, which is similarly less than that of IBRDIC ($(n+3)E+(n+1)P$) and that of FIBDIA ($(4n+1)E+(n+2)P$). In terms of communication cost, IBRDIC, FIBDIA and our scheme are similar, with costs of $|m|+2|G_1|$, $|m|+3|G|$, and $|Z_q|+3|G_1|$, respectively. Here, we consider only the user side and the TPA side since their resources are limited.

Figure 6(a) shows that our scheme has a computational overhead on the user side that is lower than that of FIBDIA but higher than that of IBRDIC. In our scheme, the user requires only 80.14 s to sign the data blocks and verify the legitimacy of the auditing results of the TPA when the number of data blocks is 10000 and the size is set to 128 bytes. The IBRDIC scheme requires only 18.82 s for this task when the number of identities is the same, but it cannot resist malicious auditors. That is, although our scheme has a slightly higher user-side computational cost for checking the validity of auditing results than IBRDIC does, our scheme is still efficient in terms of computation time while also being more practical. Figure 6(b) shows that our scheme requires the lowest computation time on the TPA side among the three schemes. In our scheme, the TPA requires only 80.11 s to audit the outsourced data when the number of data blocks is 10000. By contrast, in IBRDIC and FIBDIA, the TPA requires 179.48 s and 235.95 s, respectively, to audit the integrity of the user data when the number of identities is the same. The computation times of our scheme on both the user side and the TPA side are satisfactory for practical

Table 4 Comparison of security properties

Security	IBRDIC [32]	FIBDIA [33]	Ours
Resistance against replacement attacks	Y	Y	Y
Resistance against forgery attacks	Y	N	Y
Resistance against malicious auditors	N	N	Y

applications.

We summarize the security properties of the three schemes in Table 4. A letter Y indicates that the scheme possesses the corresponding security property, and a letter N indicates that the scheme does not have the corresponding security property. Neither IBRDIC and FIBDIA can resist malicious auditors. Moreover, FIBDIA cannot resist forgery attacks. By contrast, our scheme is resistant to both replacement attacks and forgery attacks as well as to malicious auditors because the challenge information in IBPA is based on the Bitcoin nonce. In the case of third-party auditing of data integrity, the presence of malicious auditors is a security threat that deserves consideration.

7 Conclusion

We propose the IBPA scheme, which can effectively resist malicious auditors. This approach achieves the unpredictability and traceability of challenge messages by relying on the nonces of the blockchain. Our security analysis and performance evaluation demonstrate that IBPA is safe and practical. In addition, we argue that IBPA can be regarded as a new service model in which TPAs provide users with comprehensive auditing services. Users can rely entirely on the TPAs and do not need to worry about the integrity of their data. In this model, the users' trust in cloud storage services is increased, while the user-side consumption of computational resources is minimized. Therefore, we believe that our work enhances the credibility of both cloud storage services and external auditing services. However, certain limitations remain. For example, the resistance to malicious auditors increases the computational overhead on the user side. In future work, we will further explore efficient public auditing mechanisms. First, we will look for a simpler way to construct challenge messages. In addition, simplifying the Checklog algorithm and achieving a better balance between storage overhead and communication overhead should be carefully considered.

Acknowledgements This work was supported by National Key R&D Program of China (Grant No. 2017YFB-0802000), and National Natural Science Foundation of China (Grant No. 61370203).

References

1 Wang C, Wang Q, Ren K, et al. Privacy-preserving public auditing for data storage security in cloud computing. In: Proceedings of INFOCOM, San Diego, 2010
2 Wang C, Chow S S M, Wang Q, et al. Privacy-preserving public auditing for secure cloud storage. IEEE Trans Comput, 2013, 62: 362–375
3 Ni J B, Yu Y, Mu Y, et al. On the security of an efficient dynamic auditing protocol in cloud storage. IEEE Trans Paral Distrib Syst, 2014, 25: 2760–2761
4 Ateniese G, Burns R, Curtmola R, et al. Provable data possession at untrusted stores. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, 2007. 598–609
5 Zhang Y, Xu C X, Li H W, et al. HealthDep: an efficient and secure deduplication scheme for cloud-assisted ehealth systems. IEEE Trans Ind Inf, 2018, 14: 4101–4112
6 Wang Q, Wang C, Li J, et al. Enabling public verifiability and data dynamics for storage security in cloud computing. In: Proceedings of European Symposium on Research in Computer Security, Saint-Malo, 2009. 355–370
7 Zhang J H, Dong Q C. Efficient ID-based public auditing for the outsourced data in cloud storage. Inf Sci, 2016, 343: 1–14
8 Armknecht F, Bohli J, Karame G, et al. Outsourced proofs of retrievability. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, 2014. 831–843
9 Juels A, Kaliski B. PORs: proofs of retrievability for large files. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, 2007. 584–597
10 Shacham H, Waters B. Compact proofs of retrievability. In: Proceedings of International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, 2008. 90–107

11 Worku S G, Xu C X, Zhao J N. Cloud data auditing with designated verifier. Front Comput Sci, 2014, 8: 503–512
12 Worku S G, Xu C X, Zhao J N, et al. Secure and efficient privacy-preserving public auditing scheme for cloud storage. Comput Electr Eng, 2014, 40: 1703–1713
13 Zhao J N, Xu C X, Li F G, et al. Identity-based public verification with privacy-preserving for data storage security in cloud computing. IEICE Trans Fund Electron, 2013, 96: 2709–2716
14 Liu C, Chen J J, Yang L T, et al. Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates. IEEE Trans Paral Distrib Syst, 2014, 25: 2234–2244
15 Shen J, Shen J, Chen X F, et al. An efficient public auditing protocol with novel dynamic structure for cloud data. IEEE Trans Inf Forensic Secur, 2017, 12: 2402–2415
16 Zhang Y, Xu C X, Liang X H, et al. Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation. IEEE Trans Inf Forensic Secur, 2017, 12: 676–688
17 Zhang Y, Xu C X, Li H W, et al. Cryptographic public verification of data integrity for cloud storage systems. IEEE Cloud Comput, 2016, 3: 44–52
18 Wang B Y, Li B C, Li H. Oruta: privacy-preserving public auditing for shared data in the cloud. IEEE Trans Cloud Comput, 2014, 2: 43–56
19 Wang B Y, Li B C, Li H. Panda: public auditing for shared data with efficient user revocation in the cloud. IEEE Trans Serv Comput, 2015, 8: 92–106
20 Yuan J W, Yu S C. Public integrity auditing for dynamic data sharing with multiuser modification. IEEE Trans Inf Forensic Secur, 2015, 10: 1717–1726
21 Jiang T, Chen X F, Ma J F. Public integrity auditing for shared dynamic cloud data with group user revocation. IEEE Trans Comput, 2016, 65: 2363–2373
22 Liu X M, Zhang T, Ma J F, et al. Efficient data integrity verification using attribute based multi-signature scheme in wireless network. In: Proceedings of the 5th International Conference on Intelligent Networking and Collaborative Systems, Xi'an, 2013. 173–180
23 Liu X M, Ma J F, Xiong J B, et al. Personal health records integrity verification using attribute based proxy signature in cloud computing. In: Proceedings of International Conference on Internet and Distributed Computing Systems, Hangzhou, 2013. 238–251
24 Wang Y J, Wu Q H, Qin B, et al. Identity-based data outsourcing with comprehensive auditing in clouds. IEEE Trans Inf Forensic Secur, 2017, 12: 940–952
25 Wang H Q, He D B, Tang S H. Identity-based proxy-oriented data uploading and remote data integrity checking in public cloud. IEEE Trans Inf Forensic Secur, 2016, 11: 1165–1176
26 Zhang Y, Xu C X, Yu S, et al. SCLPV: secure certificateless public verification for cloud-based cyber-physical-social systems against malicious auditors. IEEE Trans Comput Soc Syst, 2015, 2: 159–170
27 Sookhak M, Gani A, Talebian H, et al. Remote data auditing in cloud computing environments: a survey, taxonomy, and open issues. ACM Comput Surv (CSUR), 2015, 47: 65
28 Nakamoto S. Bitcoin: a peer-to-peer electronic cash system. 2008. http://www.bitcoin.org
29 Wood G. Ethereum: a Secure Decentralised Generalised Transaction Ledger. Ethereum Project Yellow Paper, 2014
30 Pilkington M. Blockchain technology: principles and applications. In: Research Handbook on Digital Transformations. Cheltenham: Edward Elgar Publishing, 2016. 225–253
31 Buterin V. On public and private blockchains. 2015. https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/
32 Yu Y, Au M H, Ateniese G, et al. Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage. IEEE Trans Inf Forensic Secur, 2017, 12: 767–778
33 Li Y N, Yu Y, Min G Y, et al. Fuzzy identity-based data integrity auditing for reliable cloud storage systems. IEEE Trans Depend Secure Comput, 2017. doi: 10.1109/TDSC.2017.2662216