Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8841711, 13 pages
https://doi.org/10.1155/2020/8841711

Research Article

A Blockchain-Based Public Auditing Scheme for Cloud Storage Environment without Trusted Auditors

Song Li,[1] Jian Liu,[1] Guannan Yang,[1] and Jinguang Han[2]

[1]*College of Information Engineering, Nanjing University of Finance and Economics, Nanjing 210003, China*
[2]*Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing 210003, China*

Correspondence should be addressed to Jinguang Han; jghan22@gmail.com

Received 24 July 2020; Revised 14 August 2020; Accepted 1 September 2020; Published 5 October 2020

Academic Editor: Kim-Kwang Raymond Choo

In the cloud storage applications, the cloud service provider (CSP) may delete or damage the user's data. In order to avoid the responsibility, CSP will not actively inform the users after the data damage, which brings the loss to the user. Therefore, increasing research focuses on the public auditing technology recently. However, most of the current auditing schemes rely on the trusted third public auditor (TPA). Although the TPA brings the advantages of fairness and efficiency, it cannot get rid of the possibility of malicious auditors, because there is no fully trusted third party in the real world. As an emerging technology, blockchain technology can effectively solve the trust problem among multiple individuals, which is suitable to solve the security bottleneck in the TPA-based public auditing scheme. This paper proposed a public auditing scheme with the blockchain technology to resist the malicious auditors. In addition, through the experimental analysis, we demonstrate that our scheme is feasible and efficient.

1. Introduction

With the rapid development of the cloud computing, users can access the cloud services more economically and conveniently today: for example, the cloud users can outsource the numerous computing tasks to the CSP and reduce the purchase of local hardware resources [1]; besides, with the help of cloud storage services such as Amazon, iCloud, and Dropbox [2], users can put aside the geographical restrictions and upload the local data to the CSP, with only a small amount of payment but a great reduction of local storage resources and more convenience of the data sharing with others. For the enterprise users, due to the explosive growth of business data, enterprises need to spend high cost to purchase software/hardware resources to build an IT system and maintain a professional technical team to manage this system, which causes extra burden to enterprises. Hence, the "pay as you go" service mode of the cloud storage is more convenient and practical. Users can dynamically apply for the storage space according to their data volume from the CSP, so as to avoid resource waste through the elastic resource allocation mechanism.

Although the cloud storage service has a broad market prospect, there are still many data security problems to be solved. Many famous CSP have experienced information disclosure and service termination [3], such as iCloud's information disclosure, Amazon cloud's storage outage, Intuit's power failure, Sidekick's cloud disaster, and Gmail's email deletion. On August 6, 2018, Tencent cloud admitted to the user's silent error caused by the firmware version of the physical hard disk; i.e., the data written is inconsistent with the data read, which damages the system metadata [4]. Therefore, solving the data integrity problem not only can enhance the user's confidence in the cloud storage services but also can effectively promote the development of the cloud storage services industry. Since cloud computing has become the basic infrastructure at the era of big data, the data security is the primary concern of cloud users.

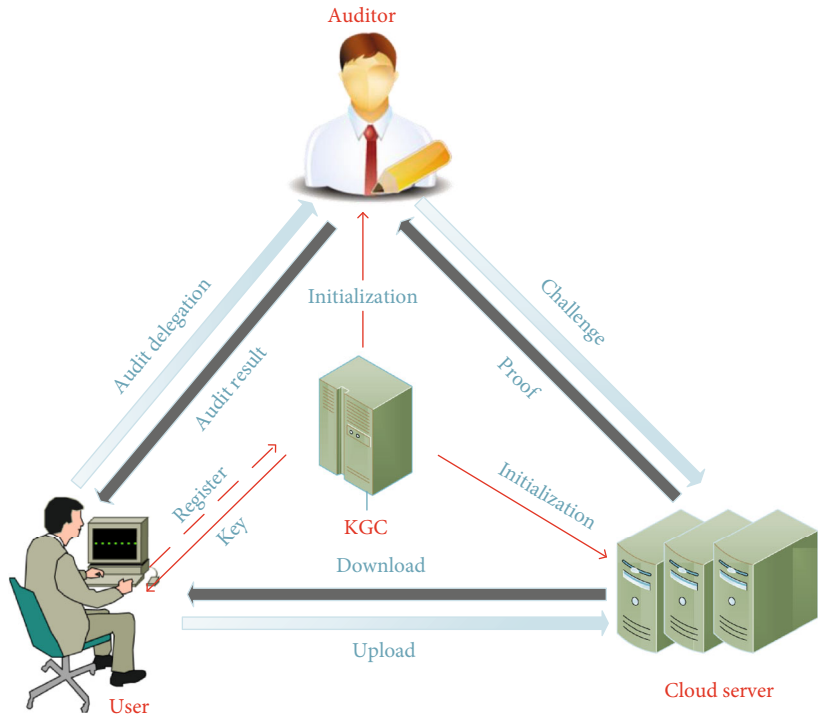

Figure 1: System model of the public auditing scheme based on the trusted third party.

However, in the practical applications, due to the system vulnerabilities, hacker attacks, hardware damage, human operation errors, or even maximizing the interests, CSP may delete or damage some user's data [5–7]. For example, the hospital outsourced all the electrical disease records to the CSP, but CSP may lose part of the stored data. It will cause a great loss to the users when these records cannot be retrieved. In order to avoid responsibility, the CSP may not actively inform the data owners after the data is damaged; in addition, in some special service models, CSP claims to provide multibackup storage service, but in the actual process, they only provide ordinary single-backup storage service and cheat the consumers to obtain additional service fees. All of these factors will cause the cloud users unable to trust the CSP fully.

The traditional method of checking the integrity of remotely stored files is to download all the data from the CSP to the local machine; then, the data owner checks it locally by computing the message authentication code or signature [8–11]. However, if the large amount of data has been stored in the remote cloud server, such as for the online retailer like Amazon that produced the hundreds of PB data every day, it is unrealistic to download all these data to the local machines every time when checking the integrity, because this will cause a lot of bandwidth/storage resources waste; on the other hand, the integrity checking is a periodic task, and it is expensive for mobile devices with limited resources to execute locally [12]; for the fairness at last, it is not reasonable to let either part of the CSP or data owners audit after the data corruption, so it is an ideal choice to introduce a trusted third party to replace CSP or data owners to check the data integrity [13] (Figure 1). In this model, the client sends a request to the auditor for auditing delegation; then, the auditor executes a challenge and response protocol to check the integrity. At last, the auditor gets the auditing result and sends it to the client. However, after the third-party auditor (TPA) has been introduced, the problem of privacy disclosure is also produced. For example, the malicious auditor obtains the data owner's identity information in the auditing process, so as to know which part of the stored data is more valuable to the user [14]; in addition, it is possible for the TPA to know the content of the stored data block in the interaction with CSP [15].

2. Related Works

In 2003, Deswarte and Quisquater [8] proposed a remote data integrity checking scheme based on the challenge-response protocol for the distributed system. Although their scheme does not need to download all the data when checking the remotely stored data, their scheme causes a large number of modular exponential operations on the server side resulting in large computing overhead; besides, the client needs to maintain all the data backup locally. In 2004, Sebe et al. [9] proposed a remote integrity checking scheme based on the Diffel-Hellman protocol. In their scheme, the client needs to store n-bits data for each data block to be stored, that is to say, only when the size of the data block is much larger than n that their scheme has practical significance (otherwise, it is not better than storing all the data locally). In 2005, Oprea and Reiter [10] proposed a scheme based on the tweakable encryption. However, the client needs to download all the files in the checking phase, and their scheme aims at data retrieval, which is not suitable for the scenario of

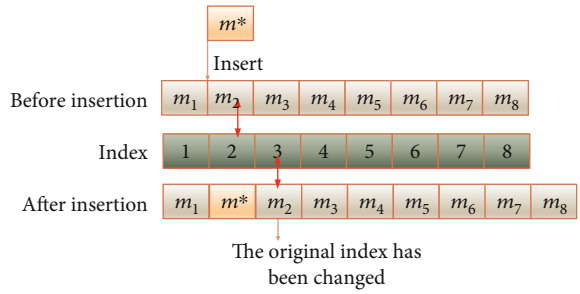

Figure 2: The invalidity of authenticators caused by the data dynamic operation (insertion).

data integrity checking. In 2006, Schwarz and Miller [11] solved the data security problem of remote storage across multiple servers based on algebraic signature. However, the computation cost in the client side increases dramatically with the increasing of the data blocks to be checked.

The proposed schemes introduced above have the same problem: the client needs to access the complete data backup; however, it is not suitable in practice obviously as mentioned before. Many scholars have carried out research on this issue later. In 2007, Ateniese et al. [16] proposed the concept of provable data possession (PDP) firstly based on RSA homomorphic linear authenticator and random sampling technology. The user can check the data stored in the remote server without downloading all the data to the local machine thus solving the defect existed in the early proposed schemes; however, their scheme only supports the static data. In 2008, Shacham and Waters proposed two improved schemes based on BLS short signature [17]: the first scheme based on BLS signature supports infinite time public verifications on the data; the second scheme calculates the authenticators using pseudorandom function but does not support public verification.

Except of the static data, users may also add, delete, or modify the remote data; these dynamic operations will change the index of the data block resulting in the invalidity of the original authenticators, as shown in Figure 2. If all the authenticators are recalculated each time when the data owner performs dynamic operations, a lot of computing and communication cost will be produced. Therefore, many scholars studied the dynamic data-supported schemes. In 2008, Ateniese et al. [18] proposed the dynamic PDP scheme based on symmetric key firstly. However, for the reason that their scheme is based on symmetric encryption, it does not support public auditing. In reference [19], Erway et al. introduced a dynamic PDP scheme that can support dynamic data using rank-based skip list technology. In reference [20], Zhu et al. proposed a scheme with an indexing-hash table to support the effective update of the dynamic data.

In 2011, Hao et al. [21] expanded the scheme of Sebe et al.'s scheme [9] and proposed a dynamic auditing scheme in block level based on RSA homomorphic tag. The so-called block level dynamic means that the data owners can insert, delete, or update data blocks, but after the update, they still need to recalculate the authenticators which is not flexible.

In the practical applications, the integrity checking task is performed by the TPA and most of the schemes proposed later support public auditing. In 2009, Wang et al. [13] proposed an integrity checking scheme with the TPA firstly based on BLS short signature and MHT (Merkle hash tree). In this scheme, any entities in the network can challenge the CSP to check the integrity of the data stored on the cloud server, but this scheme does not support the full dynamic operations on the data.

Although the introduction of the TPA brings many benefits, it also brings new security and privacy issues. Therefore, the public auditing scheme supporting privacy preserving has become a hotspot recent years. In 2010, Wang et al. [14] proposed a public auditing scheme supporting content privacy preservation based on the random mask technology. This scheme supports batch verification of multiuser tasks. However, due to the large number of verification tags generated on the server side, the system suffers a large storage burden. In 2012, Wang et al. [15] proposed a public auditing scheme to protect the identity privacy of the group users based on group signature technology, but the group signature produced huge computing cost in the data owner's side, and their scheme did not consider the situation that the users can leave and join the group dynamically. In their scheme, users need to recalculate the authenticators of all the stored data block when the group key has changed; in 2014, Wang et al. [22] proposed an auditing scheme based on ring signature technology, which can protect the identity privacy of group membership and support group members to join/leave the group dynamically, but the efficiency of their scheme is decreased with the increasing number of the group members, and the malicious users cannot be tracked in their scheme.

In the process of authenticator generation phase, a large number of signature operations are involved; however, many of the existing terminal equipment are embedded devices with low-power capacity such as mobile phones or sensors in IoT applications; therefore, public auditing schemes for low-power equipment have also been studied: in 2015, He et al. [23] proposed a public auditing scheme based on the certificateless cryptosystem and applied it into the cloud-assisted wireless body area networks. Based on their certificateless mechanism, certificates do not need to be transferred and stored compared with the previous proposals thus reducing the bandwidth resources; the users do not need to do the CRL (certificate revocation list) querying which greatly saves the computing resources. In 2016, Li et al. [12] proposed two auditing schemes for low-performance equipment based on online-offline signature technology. In the first basic scheme, the TPA needs to store some offline signature information, so it is only suitable for users to upload some short data (such as a phone number) in the cloud; in the second scheme, the author solved the problem that the TPA needs to store a large number of offline signatures.

In 2017, Li et al. [24] pointed out that most of the existing schemes are based on the PKI infrastructure and the security of these schemes depends on the security of the key and then proposed a public auditing scheme based on fuzzy identity signature technology. In this scheme, the user's identity (ID) is the public key, which improves the security of the

system. However, Xue et al. [25] pointed out that Li et al.'s scheme cannot resist a malicious auditor's attack; Yu and Wang put forward a scheme to resist key disclosure attack in the literature [26], which guarantees the forward security of the system by supporting the key updating mechanism, and the updated keys can still audit the previous data block tagged with the old keys.

In 2013, Liu et al. [27] proposed a public auditing scheme based on the rank-based Merkle-hash tree to improve the efficiency of the traditional hash tree algorithm. However, this algorithm causes a lot of computation cost to the TPA. If there are a large number of data blocks, the TPA needs to spend a lot of time to calculate the path of the Merkle tree. Yang and Jia [28] proposed a scheme based on index table structure and BLS signature algorithm, which supports the PDP mechanism of full dynamic data operation. In their scheme, because the index table is used to store the metadata of block file through a continuous storage space, the deletion and insertion move a large number of data. With the expansion of user data scale and the increase of the number of block files, the time cost of deletion and insertion will increase dramatically, which directly leads to the increasing of verification time cost after dynamic operation and reduces the auditing efficiency. In 2016, Li et al. [29] proposed that a PDP auditing model based on the LBT structure (large branching tree proofs of data possession, LPDP) to solve the problem of the authentication path is too long in building the MHT. LBT adopts a multibranch path structure, and the depth of the LBT to be constructed decreases with the increasing of out-degree, thus reducing the auxiliary information in the process of data integrity checking, simplifying the process of data dynamic update, and reducing the calculation overhead between entities in the system. In 2017, Garg and Bawa [30] added indexes and timestamps to the MHT structure introduced in the scheme [13] and proposed a rist-MHT (relative indexed and time-staged Merkle hash tree) structure. Based on this structure, they proposed a PDP mode. Compared with the MHT structure, the rist-MHT structure shortens the authentication length in MHT, thus reducing the time cost of node query. On the other hand, time stamp attribute gives the authenticator data freshness. However, although these algorithms based on MHT hash tree [13, 27, 30] avoid downloading all the data in the auditing process, the correct verification results can only prove that the cloud server stores the hash tree but not the uploaded data.

In recent years, many scholars have carried out researches on the other issues such as group user revocation, data deduplication, sensitive information sharing, and anti-quantum attack.

In 2020, Zhang et al. [31] pointed out that in the existed group sharing schemes, user revocation results in the large computational cost of the authenticator associated with the revoked users, so they proposed an identity-based public auditing scheme that can support user revocation, in which the revoking of malicious user does not affect the auditing of the previous data blocks.

Young et al. [32] combined the ciphertext deduplication technology [33] with a public auditing scheme. Because a large number of data uploading work are transferred to the CSP, the client only needs to carry out a single tag calculation step, which is suitable for a low-performance client environment.

Shen et al. [34] proposed a public auditing scheme that can hide sensitive information when the data owner was sharing the data with other users based on IBE (identity-based encryption). In this scheme, the role of data transfer (sanitizer) is added to transfer the sensitive data and its signature to realize the privacy preservation of the sensitive information in a shared medical record.

In 2019, Tian et al. [35] pointed out that up to now, none of the schemes above can meet all the security properties and put forward a new scheme. In the process of tagging, the user's signatures will be converted into group signatures, thus protecting the identity privacy of the users; in the auditing process, the content privacy is protected by using mask technology; all data operations will be recorded in the operation history table so that all illegal activities can be tracked.

Xue et al. [25] proposed a public auditing scheme based on blockchain to resist malicious auditors. In their scheme, the challenge verification information is generated based on a bitcoin algorithm. However, the final auditing result of their scheme still relies on TPA uploading to the blockchain, which does not eliminate the threat of malicious TPA fundamentally.

Through the analysis above, we can see that the proposed schemes have the following defect present: the security of these schemes relies on the trusted third party—TPA. Although the TPA brings advantages of the fairness and efficiency to the auditing process, it cannot get rid of the possibility of the malicious auditor, because there is no completely trusted third party in the real world. Although some scholars have conducted research on privacy protection problem in *TPA* based on public auditing schemes with group signature, ring signature, and other privacy protection technologies, the *TPA* needs to be treated as a semi-trusted entity and the risk of malicious auditor has not be eliminated fundamentally. As a new technology, blockchain technology can effectively solve the trust problem among multiple individuals, which is suitable to solve the security bottleneck problem in the TPA-based public auditing scheme. This paper intends to solve the malicious auditor problem in the public auditing schemes combined with blockchain technology.

Contributions. The main contributions are summarized as follows:

(1) We propose a framework of public auditing scheme without a trusted third party based on blockchain and give a basic work-flow

(2) We propose a certificateless public auditing scheme based on the proposed framework to resist the malicious auditor and key escrow problems

(3) We present a detailed security analysis of our schemes. The efficiency and security comparison shows that our scheme is better than existing schemes

3. Preliminaries

Definition 1. Bilinear map.

Given a cyclic multiplicative group G with order q and another multiplicative cyclic group G_T with the same order q, a bilinear pairing refers to a map e: $G \times G \longrightarrow G_T$ which satisfies the following properties:

(1) Bilinearity: For all $P, Q \in_R G$ and $a, b \in_R \mathbf{Z}_\mathbf{q}^*$, $e(aP, bQ) = e(P, Q)^{ab}$.

(2) Nondegeneracy: There exist $P, Q \in_R G$ such that $e(aP, bQ) \neq 1_{G_T}$.

(3) Computability: For all $P, Q \in_R G$, there exists an efficient algorithm to compute $e(aP, bQ)$.

Definition 2. Elliptic Curve Discrete Logarithm Problem (ECDLP).

Suppose that $P, Q \in_R G$. Given P and Q, it is computationally infeasible to find out the integer $s \in \mathbf{Z}_\mathbf{q}^*$ such that $Q = s \cdot P$.

Definition 3. Computational Diffel-Hellman Problem (CDHP).

Suppose that $P, Q \in_R G$ and $a, b \in_R \mathbf{Z}_\mathbf{q}^*$, it is computationally infeasible to output the result $Q = a \cdot b \cdot P$ only with $\{P, a \cdot P, b \cdot P\}$.

4. The Framework of Our Public Auditing Scheme Based on Blockchain

4.1. System Model. In our proposed framework, there are four roles: cloud server provider (CSP), client, key generating center (KGC), and auditors.

4.1.1. Cloud Service Provider. In our scheme, the CSP is a semitrusted entity with strong computing/storage resources, and the client uploads the local data to the remote CSP for storage. The CSP faithfully follows the whole process of the auditing protocol with the other entities; however, he/she attempts to cover up the fact of data corruption.

4.1.2. Client. The client is a cloud storage service user. He/she stores his/her data in the CSP to reduce the storage burden locally. To ensure the integrity of the remotely stored data, the client can delegate the auditor to execute the interactive protocol with the CSP and get the auditing result from the auditor.

4.1.3. KGC. The KGC is a trusted entity in our proposal and generates the public parameters of the whole system and the client's partial secret key in the certificateless cryptosystem.

4.1.4. Auditor. Auditors are distributed nodes deployed on the blockchain nodes, and the ProofVerify algorithm is deployed on the auditors as the form of smart contract. After getting the proof generated by the CSP, the auditors calculate the checking result and store them into the storage layer of blockchain.

The relationship among these entities is shown in Figure 3.

4.2. The Proposed Framework. In this section, we proposed a basic framework of public auditing scheme based on blockchain technology and give a general work flow. In our framework, in order to solve the problem of malicious attackers in the traditional TPA-based schemes, we use the distributed nodes in the blockchain network as auditors to check the integrity.

Before the client uploads the data to the CSP, it uses the private key issued by the KGC to calculate the linear authenticator of the file. The calculation process divides the file into data blocks for calculation firstly, and then the user uploads the data and the corresponding linear authenticator to the CSP for storage. When the client wants to check the integrity of the stored data in the cloud, the client sends the challenge information (randomly generated integers) and sends it to the auditors and CSP; the CSP calculates the proof according to the challenge information and returns the proof to the auditors.

Auditors are smart contracts deployed on the blockchain nodes, the function of which mainly includes two parts: processing client auditing request and executing the ProofVerify algorithm (the main part of the auditing scheme). The distributed auditors calculate the auditing results according to the proof returned by the CSP, store the results into the storage layer of the blockchain, and maintain a history that cannot be tampered.

Secondly, when the client performs the data updating operations (such as adding, deleting, querying, and modifying) on the stored data, the CSP generates the client's operation log of this time and compute the multiple signatures on this log by the client and CSP which indicate that all members agree with this result. It should be noted that auditing is a periodic process; it can be arranged every day at a certain fixed period such as after zero clock, but each time the user performs an updating operation, an auditing action will be triggered automatically.

If the client or CSP finds out the stored data has been damaged, they can compare the current auditing results with the previous historical records stored in the blockchain and combine the signed operation logs to determine the responsibility for data damage; because these data are stored in the distributed ledger with nonrepudiation and nontampering, neither party can refuse to admit it.

4.3. Consensus Mechanism of the Distributed Auditing Nodes. When a client sends an auditing request to the distributed auditors, the blockchain network triggers a consensus mechanism, and the data stored in the CSP is audited and stored among the nodes. We build two consensus mechanisms as shown in Figure 4: one is a secure model, and the other one is an efficient model. The following steps show the consensus mechanism between distributed auditors in the auditing process:

(1) Users broadcast the auditing requests with challenge information to the blockchain network, and the auditors store the challenge information

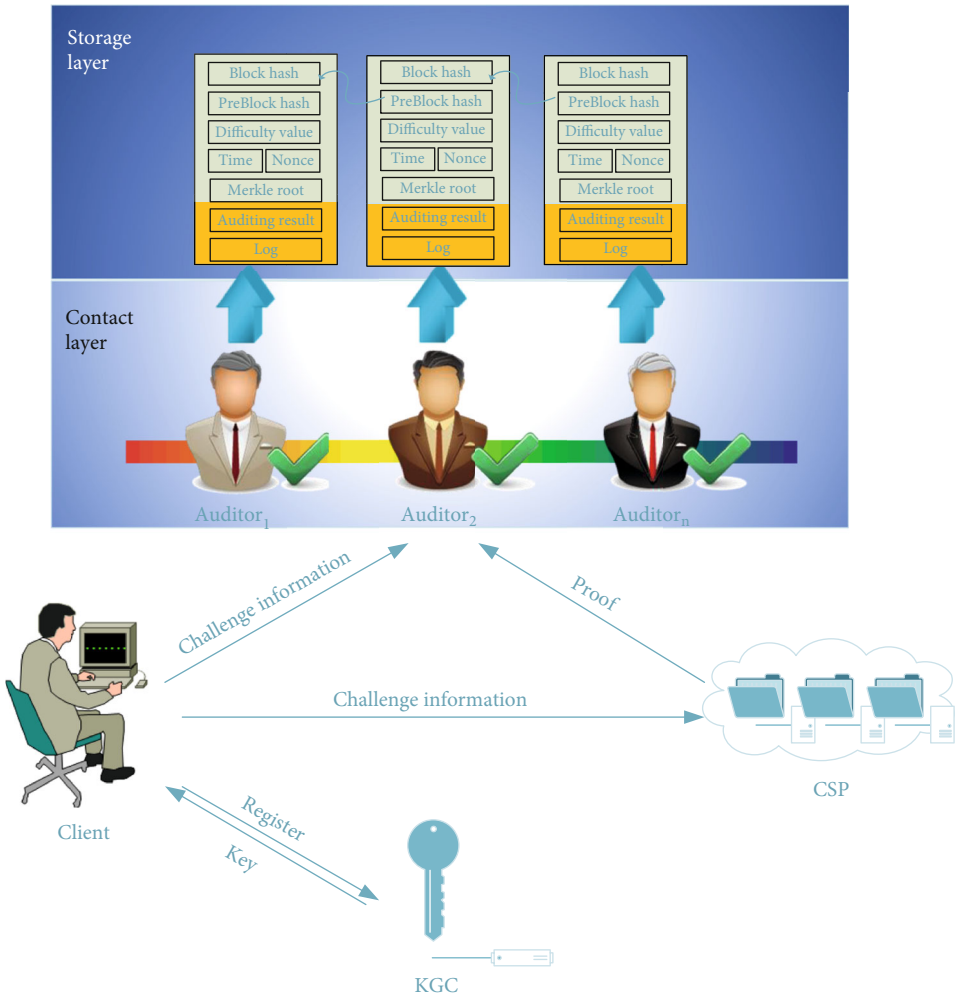

Figure 3: The proposed framework against malicious auditors for cloud storage based on the blockchain.

(2) The two mechanisms are different from this step. In the efficient mechanism, when the CSP receives the auditing requests, the CSP divides the data into n parts according to the number of auditing nodes to be received and sends them to different auditors; in the secure mechanism, the CSP does not divide the data into parts but broadcast them to the network and all the distributed nodes can get all the data blocks

(3) After receiving the data blocks, each auditor executes the ProofVerify algorithm with the input of the user's public key and the proof sent from the CSP; in the efficient model (the left one in Figure 4), the auditing task is divided into parts and the auditors only audit partial data blocks to improve the auditing speed; in the secure mechanism (the right one in Figure 4), each auditor audits all the data blocks; therefore, it can resist the attacks from the single malicious auditor

(4) Finally, the auditors store the auditing result with the following steps: in the efficient model, the auditors broadcast the auditing result to the other nodes in the same blockchain network, and all the storage nodes can get the full auditing results of the entire request data blocks; in the secure model, the auditors do not need to broadcast the auditing result in the network.

5. The Detailed Scheme

In this section, we give a detailed proposal based on the framework we introduced above. Our scheme is constructed based on Li et al.'s CLPA [24] scheme and Yu and Wang's scheme IDBA [26].

(1) **Setup**: with input in the security parameter κ, the KGC generates the system parameters and the master key executes the following steps:

(1) The KGC selects a large prime number q, an additive group G_1, and uses the bilinear group generator to generate the bilinear group G_2; normally, G_1 and G_2 can be generated simultaneously by using the bilinear group generator. The KGC chooses a bilinear pairing $e : G_1 \times G_1 \longrightarrow G_2$

(2) Let P be a generator of group G_1. The KGC selects a big integer $s \in \mathbf{Z}_q^*$ randomly as the master key, keeps s secretly, and computes the public key $P_{pub} = sP$

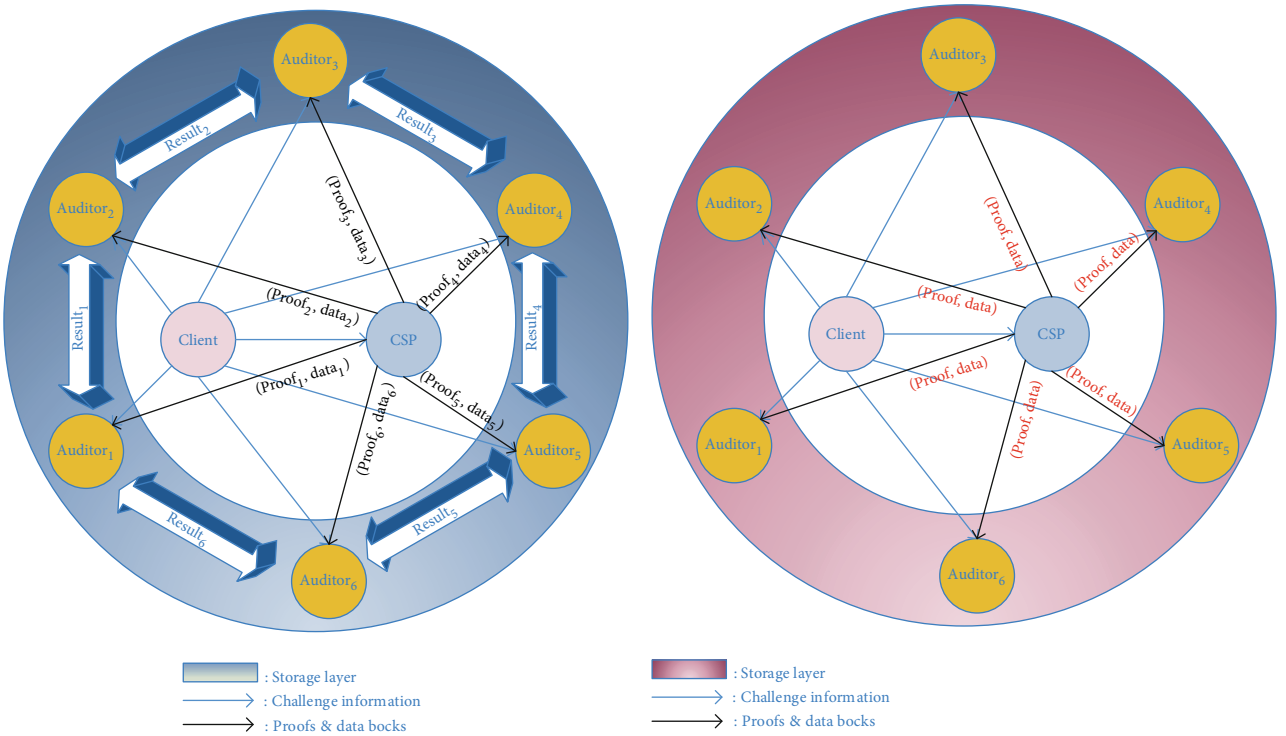

Figure 4: The proposed consensus mechanism between the distributed auditing nodes in two different models.

(3) The KGC publishes the system parameters $\text{Para} = \{q, G_1, G_2, P, e, h_1(\cdot), h_2(\cdot), h_3(\cdot), H_1(\cdot), H_2(\cdot), P_{\text{pub}}\}$, where $h_1(\cdot), h_2(\cdot), h_3(\cdot), H_1(\cdot), H_2(\cdot)$ are five hash functions

(2) **PartialPrivateKeyExtract**: the client registers with the KGC to extract the partial private key with the following steps:

(1) The client submits his/her identity ID_U to the KGC

(2) After receiving the client's identity ID_U, the KGC chooses a random big integer $t_U \in \mathbf{Z}_{\mathbf{q}}^*$ and computes $T_U = t_U \cdot P$, $h_U = h_1(\text{ID}_U, T_U)$ and $s_U = t_U + s \cdot h_U$ mode q

(3) The *KGC* sends the partial private key $D_U = \{s_U, T_U\}$ to the user secretly

(3) **SetSecretValue**: the client sets his/her secret value as follows:

(1) The client chooses a big integer x_U randomly as his/her secret value

(2) The client keeps x_U secretly

(4) **SetPublicKey**: the client sets his/her public key as follows:

(1) The clients computes $P_U = x_U \cdot P$

(2) The clients sets $pk_U = \{T_U, P_U\}$ as his/her public key

(5) **SetPrivateKey**: the client sets $ssk_U = \{s_U, x_U\}$ as his/her private key.

(6) **Store**: the client O with identity ID_O, private key $ssk_O = \{s_O, x_O\}$, and public key $pk_O = \{T_O, P_O\}$ runs this algorithm to generate the integrity checking tags for the data file F. Firstly, the data file F should be divided into n blocks $\{m_1, m_2, \cdots, m_n\}$; for every data blocks $m_i, i \in \{1, 2, \cdots, n\}$, the client computes the tags with the following steps:

(1) The client computes $k_O = h_2(\text{ID}_O, pk_O, P_{\text{pub}})$ and $Q = H_1(P_{\text{pub}})$

(2) The client computes $S_i = (s_O + k_O \cdot x_O)(r \cdot H_2(m_i) + H_2(\text{id}_i) + m_i \cdot Q)$ and sends $\{m_i, \text{id}_i, S_i, R\}$ to the CSP, where id_i is the unique identity of m_i and r is a random number

$$R = r \bullet \left(T_O + h_O \bullet P_{\text{pub}} + k_O \bullet P_O\right). \quad (1)$$

(7) **Audit**: to check the integrity of the uploaded data, the client executes the following challenge-response protocol with CSP and auditors:

(1) *Challen*: the client generates a challenge information as follows:

(i) Selects a random l-element subset $J = \{a_1, a_2, \cdots, a_l\}$ of the set $[1, n]$

(ii) Selects a random $v_j \in \mathbf{Z}_{\mathbf{q}}^*$ for each $j \in J$

(iii) Generates the challenge information: $Chall = \{j, v_j\}_{j\in J}$ and broadcasts it in the network; CSP and all the auditors can get it

(2) *ProofGen*: after receiving the challenge information $Chall = \{j, v_j\}_{j\in J}$ from the client, the CSP generates a proof which proves the correctly possession of selected blocks as follows:

(i) Chooses a big integer $x \in \mathbf{Z}_{\mathbf{q}}^{*}$ randomly

(ii) Computes

$$u = x^{-1} \cdot \left(\sum_{j=a_1}^{a_1} m_j \cdot v_j + h_3(\sigma) \right), \tag{2}$$

$$\sigma = x \cdot Q \in G_1, \tag{3}$$

$$\delta = \sum_{j=a_1}^{a_1} v_j \cdot S_j \tag{4}$$

(iii) Broadcasts the proof information $Prof = \{\delta, u, \sigma, R\}$ to the auditors; if the client chooses to audit in the efficient model, the CSP needs to divide the data blocks into k parts and generate the proof information for every set of data blocks; then, the CSP sends them to the k auditors separately

(8) **ProofVerify**: upon receiving the $Prof = \{\delta, u, \sigma, R\}$, the auditors execute this algorithm to check the integrity of the data stored in the CSP. Here, the *Prof* indicates the proof generated by the CSP; in the secure model, the *Prof* is the proof information of all the data blocks; while in the efficient model, the *Prof* is the partial proof information. We use the same expression as the *Prof* here.

(1) The auditors compute $h_O = h_1(\mathrm{ID}_O, T_O)$, $k_O = h_2(\mathrm{ID}_O, pk_O, P_{\mathrm{pub}})$, and $Q = H_1(P_{\mathrm{pub}})$

(2) The auditors check whether the following equation holds

$$\begin{aligned} e(\delta, P) = {} & e\left(\sum_{j=a_1}^{a_1} v_j \cdot H_2(\mathrm{id}_j), T_O + h_O \cdot P_{\mathrm{pub}} + k_O \cdot P_O \right) \\ & \cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot H_2(m_j), R \right) \cdot e(u\sigma - h_3(\sigma)Q, T_O \\ & + h_O \cdot P_{\mathrm{pub}} + k_O \cdot P_O). \end{aligned} \tag{5}$$

If it is, the auditors output 1 to indicate the correct storage of the data File F; otherwise, the auditors output 0 to indicate data corruption

(3) The auditors create an **entry**$(t, nonce, Chall, Prof, 0/1)$ and broadcast it in the network, and all the auditors can get the full auditing result and store them; in the secure model, each auditor can calculate the full auditing result by themselves, and the broadcast operation is not needed

(9) **DataUpdate**: when the client updates the file in the cloud, a recording log *Log* is generated by the CSP to record the details of the client's operation. The CSP and client execute the *MultiSign*(*Log*) and broadcast it in the blockchain network for storage, the *MultiSign*(*Log*) means the multisignature of the client and the CSP on the *Log*. After each data **DataUpdate** operation finished, the system automatically triggers the **Audit** phase.

6. Security Analysis and Correctness Proof

This section gives the correctness proof and security analysis of our proposed scheme. We mainly introduced the threat model and discussed the security goals which we have achieved in this part.

6.1. Correctness Proof. The correctness of our auditing scheme can be derived as follows:

$$\begin{aligned} e(\delta, P) &= e\left(\sum_{j=a_1}^{a_1} v_j \cdot S_j, P \right) = e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \right. \\ &\quad \left. \cdot \left(r \cdot H_2(m_j) + H_2(\mathrm{id}_j) + m_j \cdot Q \right), P \right) \\ &= e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \cdot \left(r \cdot H_2(m_j) + v_j \cdot (s_O \right. \right. \\ &\quad \left. \left. + k_O \cdot x_O) \cdot H_2(\mathrm{id}_j) + v_j \cdot (s_O + k_O \cdot x_O) \cdot m_j \cdot Q, P \right) \right. \\ &= e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \cdot r \cdot H_2(m_j), P \right) \\ &\quad \cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \cdot H_2(\mathrm{id}_j), P \right) \\ &\quad \cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \cdot m_j \cdot Q, P \right) \\ &= e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \cdot r \cdot H_2(m_j), P \right) \\ &\quad \cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \cdot H_2(\mathrm{id}_j), P \right) \\ &\quad \cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot (s_O + k_O \cdot x_O) \cdot m_j \cdot Q, P \right) \\ &= e\left(\sum_{j=a_1}^{a_1} v_j \cdot H_2(\mathrm{id}_j), T_O + h_O \cdot P_{\mathrm{pub}} + k_O \cdot P_O \right) \end{aligned}$$

$$\cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot H_2(m_j), R\right) \cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot m_j \cdot Q, T_O + h_O \cdot P_{\text{pub}} + k_O \cdot P_O\right) = e\left(\sum_{j=a_1}^{a_1} v_j \cdot H_2(\text{id}_j), T_O + h_O \cdot P_{\text{pub}} + k_O \cdot P_O\right) \cdot e\left(\sum_{j=a_1}^{a_1} v_j \cdot H_2(m_j), R\right) \cdot e(U \cdot \sigma - h_3 \cdot (\sigma) \cdot Q, T_O + h_O \cdot P_{\text{pub}} + k_O \cdot P_O). \tag{6}$$

To this step, we can see that through the verification of Equation (5), the auditors can check the integrity of the stored data in the CSP correctly.

6.2. Threat Model. Before our security proof, we introduce the threat model of our scheme in this part firstly. Similar to the literature [26], we consider that there are three types of attacks in the public auditing schemes: forgery, replacement, and replay attacks. Each type of the attack is defined as follows:

(1) Replacement attack: the adversary attempts to calculate a new block/signature passing the auditing phase by replacing the challenged block and signature with unchallenged or uncorrupted blocks/signatures.

(2) Forgery attack: adversary forges the proof information to deceive the auditor/user or forges an auditing result to cheat the user.

(3) Replay attack: adversary replays the proof information generated previously attempting to pass the auditing phase.

Similar to the literature [26], we consider that the CSP may launch all the attacks above and the auditor may launch forgery attack. In addition, we consider that external adversaries may launch forgery and replay attacks.

6.3. Security Proof

Theorem 4. *Our scheme can resist replacement attacks from the CSP.*

Proof. Suppose that the CSP wants to use the well-maintained data blocks m_{k_1} and m_{k_2} to replace the corrupted block m_k in the file F, where $k, k_1, k_2 \in [1, n]$. During the auditing process, both the auditors and the client execute the protocol honestly. That is, the client computes $S_i = (s_O + k_O \cdot x_O) \cdot (r \cdot H(m_i) + H(\text{id}_i) + m_i \cdot Q)$ in the store phase.

Then, the client sends the tags $\{m_i, \text{id}_i, S, R\}$ to the CSP. We denote $(s_O + x_O \cdot k_O)$ as ω here:

Since

$$S_{k_1} = \omega \cdot \left(r \cdot H_2(m_{k_1}) + H_2(\text{id}_{k_1}) + m_{k_1} \cdot Q\right), \tag{7}$$

$$S_{k_2} = \omega \cdot \left(r \cdot H_2(m_{k_2}) + H_2(\text{id}_{k_2}) + m_{k_2} \cdot Q\right), \tag{8}$$

it follows that

$$\begin{aligned} S_k^* &= \alpha_{k_1} \cdot S_{k_1} + \alpha_{k_2} \cdot S_{k_2} = \alpha_{k_1}\omega \cdot \left(rH_2(m_{k_1}) + H_2(\text{id}_{k_1}) + m_{k_1} \cdot Q\right) + \alpha_{k_2}\omega \cdot \left(r \cdot H_2(m_{k_2}) + H_2(\text{id}_{k_2}) + m_{k_2} \cdot Q\right) \\ &= \omega\left(\left(\alpha_{k_1} \cdot \left(r \cdot H_2(m_{k_1}) + H_2(\text{id}_{k_1}) + m_{k_1} \cdot Q\right) + \alpha_{k_2} \cdot \left(r \cdot H_2(m_{k_2}) + H_2(\text{id}_{k_2}) + m_{k_2} \cdot Q\right)\right)\right. \\ &= \omega\left(\left(\alpha_{k_1} \cdot \left(H_2(\text{id}_{k_1}) + \alpha_{k_2} \cdot \left(H_2(\text{id}_{k_2})\right)\right)\right.\right. \\ &\quad \left.\left. + \left(\alpha_{k_1} \cdot m_{k_1} + \alpha_{k_2} \cdot m_{k_2}\right) \cdot Q + r \cdot \left(\alpha_{k_1} \cdot H_2(mk_1) + \alpha_{k_2} \cdot H_2(m_{k_2})\right)\right)\right). \end{aligned} \tag{9}$$

We know that if the S_k^* can pass the verification phase, the following equation must hold:

$$\begin{aligned} S_k^* &= \omega\left(\left(\alpha_{k_1} \cdot \left(H_2(\text{id}_{k_1}) + \alpha_{k_2} \cdot \left(H_2(\text{id}_{k_2})\right)\right) + \left(\alpha_{k_1} \cdot m_{k_1} + \alpha_{k_2} \cdot m_{k_2}\right) \cdot Q + r \cdot \left(\alpha_{k_1} \cdot H_2(m_{k_1}) + \alpha_{k_2} \cdot H_2(m_{k_2})\right)\right)\right) \\ &= \omega(H_2(\text{id}_k) + m_k \cdot Q + r \cdot H_2(m_k)). \end{aligned} \tag{10}$$

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

$$\alpha_{k_1} \cdot \left(H_2(\text{id}_{k_1}) + \alpha_{k_2} \cdot H_2(\text{id}_{k_2})\right) = H_2(\text{id}_k), \tag{11}$$

$$\alpha_{k_1} \cdot m_{k_1} + \alpha_{k_2} \cdot m_{k_2} = mk, \tag{12}$$

$$\alpha_{k_1} \cdot H_2(m_{k_1}) + \alpha_{k_2} \cdot H_2(m_{k_2}) = H_2(m_k). \tag{13}$$

That is, S_k^* cannot pass the auditing of the verification phase. Therefore, our scheme can resist the CSP's replacement attacks.

Theorem 5. *Our scheme can resist forgery attacks from the CSP or the auditor.*

Proof. Suppose that the adversary modifies the data block m_k to $m * k = m_k + l_k, k \in [1, n]$. During the auditing process, both the auditors and the CSP honestly execute the scheme. That is, in the **Audit** phase, the client broadcasts the challenge message $Chall = \{j, v_j\}_{j \in J}$ to the CSP and auditors in the network. In the *ProofGen* phase, the CSP computes the following steps:

$$\tau \sum_{k=1}^{n} (m_k + l_k) \cdot v_k, \tag{14}$$

$$\begin{aligned}\widehat{u} &= x^{-1}(\tau + h_3(\sigma)) = x^{-1}\left(\sum_{k=1}^{n}(m_k + l_k)\cdot v_k + h_3(\sigma)\right)\\ &= x^{-1}\left(\sum_{k=1}^{n} m_k \cdot v_k + \sum_{k=1}^{n} l_k \cdot v_k + h_3(\sigma)\right)\\ &= x^{-1}\sum_{k=1}^{n} m_k \cdot v_k + x^{-1}\cdot \sum_{k=1}^{n} l_k \cdot v_k + x^{-1}\cdot h_3(\sigma)\\ &= u + x^{-1}\cdot \sum_{k=1}^{n} l_k \cdot v_k.\end{aligned} \tag{15}$$

If the modified tag $\widehat{u}$ can be passed in the verification phase, the adversary must compute the following:

$$\Delta u = \widehat{u} - u = x^{-1}\cdot \sum_{k=1}^{n} v_k \cdot l_k. \tag{16}$$

Note that x is randomly selected by the CSP and that v_k is randomly selected by the client, so the x and v_k cannot be known simultaneously by the same adversary; therefore, the adversary's modified tag cannot be passed in the **ProofVerify** phase. Hence, our scheme can resist the forgery attacks from the CSP or the auditor.

Theorem 6. *Our scheme can resist replay attack from the CSP.*

Proof. If the stored data m_k has been corrupted, the CSP may attempt to pass the auditing phase by replaying another block m_i and its corresponding tag S_i. Then the CSP constructs the tampered proof S^* as follows: we denote $(s_O + x_O \cdot h_2(\mathrm{ID}_O, pk_O, P_{\mathrm{pub}}))$ as π here:

$$S^* = v_j S_j + \sum_{j\in J j\neq k} v_j S_j. \tag{17}$$

Then, we have the following derivation of the **ProofVerify** process:

$$\begin{aligned}e(S^*, P) &= e\left(v_j S_j + \sum_{j\in J, j\neq k} v_j S_j, P\right)\\ &= e\left(v_j\pi\left(r\cdot H_2(m_j) + H_2(\mathrm{id}_j) + m_j\cdot Q\right), P\right)\\ &\quad\cdot e\left(\sum_{j\in J, j\neq k} v_j\pi\left(r\cdot H_2(m_j) + H_2(\mathrm{id}_j) + m_j\cdot Q\right), P\right)\\ &= e\left(v_j\pi r\cdot H_2(m_j), P\right)\cdot e\left(v_j\pi H_2(\mathrm{id}_j), P\right)\\ &\quad\cdot e\left(v_j\pi m_j\cdot Q, P\right)\cdot e\left(\sum_{j\in J, j\neq k} v_j\pi r\cdot H_2(m_j), P\right)\\ &\quad\cdot e\left(\sum_{j\in J, j\neq k} v_j\pi r\cdot H_2(\mathrm{id}_j), P\right)\cdot e\left(\sum_{j\in J, j\neq k} v_j\pi m_j Q, P\right)\\ &= e\left(\pi r\cdot\left(H_2(m_j) + \sum_{j\in J, j\neq k} H_2(m_j)\right), P\right)\\ &\quad\cdot e\left(v_j\pi\cdot\left(H_2(\mathrm{id}_j) + \sum_{j\in J, j\neq k} H_2(\mathrm{id}_j)\right), P\right)\\ &\quad\cdot e\left(v_j\pi\cdot m_j\cdot Q + \sum_{j\in J, j\neq k} v_j\pi m_j\cdot Q, P\right)\\ &= e\left(v_j\pi r\cdot\left(H_2(m_j) + \sum_{j=\alpha_1}^{\alpha_1} H_2(m_j) - H_2(m_k)\right), P\right)\\ &\quad\cdot e\left(v_j\pi\cdot\left(H_2(\mathrm{id}_j) + \sum_{j=\alpha_1}^{\alpha_1} H_2(\mathrm{id}_j) - H_2(\mathrm{id}_k)\right), P\right)\\ &\quad\cdot e\left(v_j\pi\cdot\left(m_j Q + \sum_{j=\alpha_1}^{\alpha_1} m_j\cdot Q - m_k\cdot Q\right), P\right)\\ &= e\left(v_j\pi r\cdot\left(H_2(m_j) - H_2(m_k) + \sum_{j=\alpha_1}^{\alpha_1} H_2(m_j), P\right)\right.\\ &\quad\cdot e\left(v_j\pi\cdot\left(H_2(\mathrm{id}_j) - H_2(\mathrm{id}_k) + \sum_{j=\alpha_1}^{\alpha_1} H_2(\mathrm{id}_j)\right), P\right)\\ &\quad\left.\cdot e\left(v_j\pi\cdot\left(m_j\cdot Q - m_k\cdot Q + \sum_{j=\alpha_1}^{\alpha_1} m_j\cdot Q\right), P\right)\right).\end{aligned} \tag{18}$$

If the tampered proof S^* can pass the auditing phase, the following equations must hold.

$$v_j H_2(m_j) - v_j H_2(m_k) = 0, \tag{19}$$

$$v_j H_2(\mathrm{id}_j) - v_j H_2(\mathrm{id}_k) = 0, \tag{20}$$

$$v_j m_j - v_j m_k = 0. \tag{21}$$

Since the hash function $H_2(\cdot)$ is collision resistant, we know that

$$H_2(m_j) - H_2(m_k) \neq 0. \tag{22}$$

In other words, the proof shows that the CSP-generated information S^* cannot pass the auditing phase. Therefore, our scheme can resist the replay attacks.

6.4. The Other Security Requirement Discussions. This section discussed that our proposed scheme satisfies the security requirements of auditing schemes. Table 1 gives a brief security comparison of our scheme with the CLPA [23] and IDBA [25].

(1) *Publicly verifiability*: through the correctness proof part, if the client correctly calculates the data tags before uploading the data file, the auditor can perform an interactive algorithm with the CSP and get the real storage situation of the data blocks without

Table 1: The security comparison of our scheme with CLPA and IDBA.

Properties	Key escrow	Replacement attack	Replay attack	Forgery attack	Malicious auditor
CLPA [23]	√	×	×	×	×
IDBA [25]	×	√	√	√	×
Our scheme	√	√	√	√	√

Table 2: The computation cost comparison of our scheme with CLPA and IDBA.

Scheme	User's computational cost	Auditing computational cost	Communication cost
CLPA [23]	$2nT_M + (n+1)T_H + T_h$	$2T_p + (n+3)T_M + (n+1)T_H + 2T_h$	$\lvert\mathbf{Z_q}\rvert + \lvert G_1\rvert$
IDBA [25]	$3nT_M + nT_H + nT_h$	$3T_p + (2n+3)T_M + nT_H + (n+1)T_h$	$\lvert\mathbf{Z_q}\rvert + 3\lvert G_1\rvert$
Ours	$(3n+3)T_M + (2n+1)T_H + T_h$	$4T_p + ((2n+4)T_M + 2nT_H + T_h/k)$	$\lvert\mathbf{Z_q}\rvert + 3\lvert G_1\rvert$

the help of the client. Therefore, we say that our scheme achieves the property of publicly verifiability.

(2) *Privacy preserving*: in the process of the data auditing, the auditors can only get the aggregated data blocks and the tags. Based on this information, auditors cannot get any available information about stored data. Therefore, we say that our scheme achieves the goal of privacy protection.

(3) *Batch auditing*: through the derivation of the correctness analysis, in the process of the auditing phase, multiple data blocks can be sampled at one time, and multiple data auditing tasks can be batch verified to improve the auditing efficiency. Therefore, our scheme achieves the goal of the batch auditing.

(4) *Key escrow resistant:* similar to the scheme CLPA [23], our scheme is based on the certificateless cryptography; the secret key to generate the authenticator has two parts which is derived from the KGC and client, respectively. Therefore, the KGC cannot get the full of the user's secret key like the scheme IDBA [25] based on the identity cryptosystem.

(5) *Malicious auditor resistant*: in our auditing scheme, the auditing result is calculated by the distributed nodes; none of them can tamper the auditing result only if the attacker controls 51% of the nodes in the network; compared to the existing blockchain-based public auditing scheme [25], the **ProofVerify** phase is transferred to the blockchain in the form of smart contract, instead of relying on the third-party auditor to upload the auditing result to the blockchain; thus, the possibility of the auditor creating the false result is eliminated fundamentally; besides, for the reason that the data blocks are confused with the mask code and the auditors can get nothing about the auditing data, the privacy of the data content has been protected.

6.5. Experimental Analysis. This section compares the performance of our proposed scheme with those of He et al.'s CLPA [23] scheme and the scheme IDBA [25]. Table 2 shows the

Table 3: The notation list.

Symbol	The time cost of corresponding operation
T_M	The point multiplication operation in G_1
T_p	The pairing operation
T_H	Hash to point function
T_h	Hash function

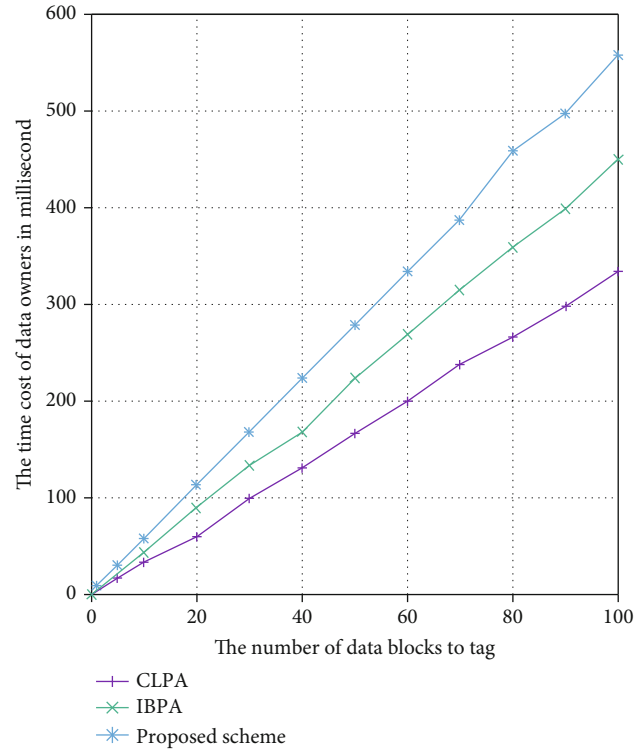

Figure 5: The computation cost on the client side versus the number of data blocks.

security overhead of these schemes in the **Store** phase on the client side and the **ProofVerify** phase on the auditors' side. From Table 2, we can see that in the **Store** phase, the time consumption of the authenticator calculation in our scheme is slightly higher than those in the other two schemes,

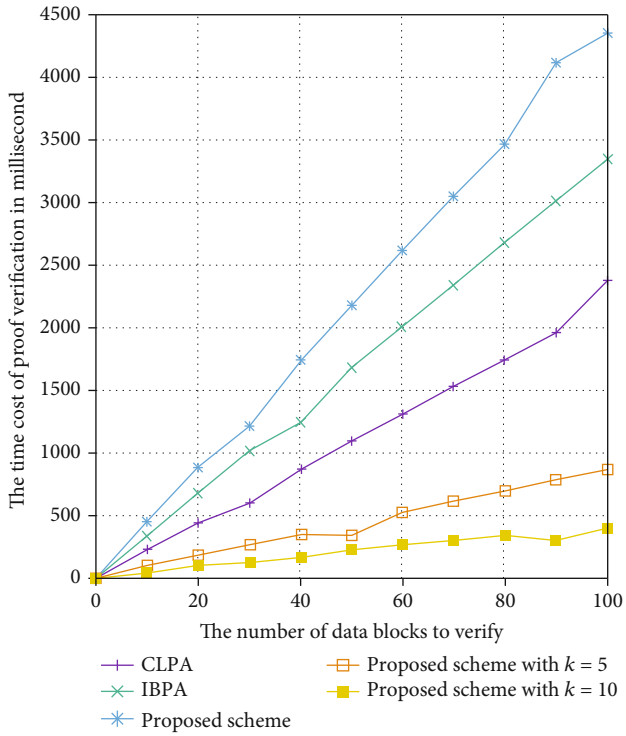

Figure 6: The computation cost on the auditor side versus the number of data blocks.

because we have done some additional processing in this phase to resist the forgery attack and replay attack in the **ProofVerify** phase.

In the **ProofVerify** stage, because we used the distributed auditors to audit the data blocks, we get better efficiency than the other schemes. We can see that if we do not use distributed auditors for auditing tasks, the computing cost of our scheme is the highest, but after using the distributed processing mechanism in the efficient model, the efficiency has been improved greatly. Table 3 is the notations list we used in Table 2.

Finally, in order to quantify this comparison, we compare these targets with the jPBC, which is a well-known JAVA cryptographic library [36]. The experimental environment is listed as follows: Intel i7 processor with 1.8 GHz clock speeds and 8G RAM in a Win 10 operation system. We compared the computational cost in the tag generation phase and the proof verifying phase in Figures 5 and 6. In the comparison of the auditing phase, we analyze the two cases of $k = 5$ and $k = 10$, where k represents the number of the distributed auditors in the blockchain network in the efficient model. We can see that in the efficient model, the more auditors we used in the blockchain network, the lower auditing delay will be obtained.

Communication Cost. In the three schemes, the challenge information is the same; in the response phase, the proof returned by our scenario is as follows: $Prof = \{\delta, u, \sigma, R\} = |\mathbf{Z_q}| + 3|G_1|$. Through the comparison of Table 2, we can find that our scheme has the same communication cost with IBDA and slightly higher than CLPA.

7. Conclusion

In this paper, we pointed out that most of the TPA-based public auditing schemes cannot resist the malicious auditor. To solve this problem, we proposed a public auditing framework with blockchain technology and certificateless cryptography. In this framework, we used the distributed nodes in the blockchain network as auditors to check the integrity and the checking results will be stored into the storage layer of the blockchain with the tamper-resistant manner; the client operations on the data will be recorded as log signed by the data owners and CSP which indicate that all members agree with this result. Anyone can check the historical records stored in the blockchain nodes and combine with the signed operation logs to determine the responsibility for data damage. We gave a detailed proven security proof of our scheme. A comprehensive performance evaluation shows that our scheme is more feasible and efficient than similar schemes.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is partially supported by the National Key Research and Development Program of China (Grant no. 2017YFD0401002-3) and the Six Talent Peaks Project in Jiangsu Province, China (Grant no. 2015-DZXX-020).

References

[1] M. Armbrust, A. Fox, R. Griffith et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] D.-G. Feng, M. Zhang, Y. Zhang, and X. Zhen, "Study on cloud computing security," *Journal of Software*, vol. 22, no. 1, pp. 71–83, 2011.

[3] W. Shen, J. Yu, H. Xia, H. Zhang, X. Lu, and R. Hao, "Light-weight and privacy-preserving secure cloud auditing scheme for group users via the third party medium," *Journal of Network and Computer Applications*, vol. 82, pp. 56–64, 2017.

[4] http://www.sohu.com/a/245553016_671058.

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

[6] D. Song, E. Shi, I. Fischer, and U. Shankar, "Cloud data protection for the masses," *IEEE Computer*, vol. 45, no. 1, pp. 39–45, 2012.

[7] A. Juels and A. Oprea, "New approaches to security and availability for cloud data," *Communications of the ACM*, vol. 56, no. 2, pp. 64–73, 2013.

[8] Y. Deswarte, J. J. Quisquater, and A. Saïdane, "Remote Integrity Checking," in *Working Conference on Integrity and*

Internal Control in Information Systems, Springer, Boston, MA, 2003.

[9] F. Sebe, A. Martinez-Balleste, Y. Deswarte, J. Domingo-Ferrer, and J. J. Quisquater, *Time-bounded remote file integrity checking*, Technical Report 04429, 2004.

[10] A. Oprea and M. K. Reiter, "Space-Efficient Block Storage Integrity," in *Network & Distributed System Security Symposium*, DBLP, 2005.

[11] T. S. J. Schwarz and E. L. Miller, "Store, forget, and check: using algebraic signatures to check remotely administered storage," in *IEEE International Conference on Distributed Computing Systems*, Lisboa, Portugal, Portugal, 2006.

[12] J. Li, L. Zhang, J. K. Liu, H. Qian, and Z. Dong, "Privacy-preserving public auditing protocol for low-performance end devices in cloud," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2572–2583, 2016.

[13] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *European symposium on research in computer security*, pp. 355–370, Springer, Berlin, Heidelberg, 2009.

[14] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *2010 Proceedings IEEE INFOCOM*, San Diego, CA, USA, March 2010.

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

[16] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrusted stores," *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007*, 2007, Alexandria, Virginia, USA, October 2007, 2007.

[17] H. Shacham and B. Waters, "Compact Proofs of Retrievability. Advances in Cryptology - ASIACRYPT 2008," Springer, Berlin Heidelberg, 2008.

[18] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, Istanbul Turkey, September 2008.

[19] C. C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," *ACM Transactions on Information and System Security (TISSEC)*, vol. 17, no. 4, pp. 1–29, 2015.

[20] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 12, pp. 2231–2244, 2012.

[21] Z. Hao, S. Zhong, and N. Yu, "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 9, pp. 1432–1437, 2011.

[22] B. Wang, B. Li, and H. Li, "Oruta: privacy-preserving public auditing for shared data in the cloud," *IEEE transactions on cloud computing*, vol. 2, no. 1, pp. 43–56, 2014.

[23] D. He, S. Zeadally, and L. Wu, "Certificateless public auditing scheme for cloud-assisted wireless body area networks," *IEEE Systems Journal*, vol. 12, no. 1, pp. 64–73, 2015.

[24] Y. Li, Y. Yu, G. Min, W. Susilo, J. Ni, and K.-K. R. Choo, "Fuzzy identity-based data integrity auditing for reliable cloud storage systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 1, pp. 72–83, 2017.

[25] J. Xue, C. Xu, J. Zhao, and J. Ma, "Identity-based public auditing for cloud storage systems against malicious auditors via blockchain," *Science China Information Sciences*, vol. 62, no. 3, article 32104, 2019.

[26] J. Yu and H. Wang, "Strong key-exposure resilient auditing for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1931–1940, 2017.

[27] C. Liu, J. Chen, L. T. Yang et al., "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, pp. 2234–2244, 2013.

[28] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 9, pp. 1717–1726, 2013.

[29] Y. Li, G. Yao, L. Lei, X. Zhang, and K. Yang, "LBT-based cloud data integrity verification scheme," *Journal of Tsinghua University (Science and Technology)*, vol. 56, no. 5, pp. 504–510, 2016.

[30] N. Garg and S. Bawa, "RITS-MHT: relative indexed and time stamped merkle hash tree based data auditing protocol for cloud computing," *Journal of Network and Computer Applications*, vol. 84, pp. 1–13, 2017.

[31] Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren, "Enabling efficient user revocation in identity-based cloud storage auditing for shared big data," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 3, pp. 608–619, 2018.

[32] T. Y. Youn, K. Y. Chang, K. H. Rhee, and S. U. Shin, "Efficient client-side deduplication of encrypted data with public auditing in cloud storage," *IEEE Access*, vol. 6, pp. 26578–26587, 2018.

[33] J. Li, J. Li, D. Xie, and Z. Cai, "Secure auditing and deduplicating data in cloud," *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2386–2396, 2016.

[34] W. Shen, J. Qin, J. Yu, R. Hao, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 331–346, 2018.

[35] H. Tian, F. Nan, H. Jiang, C. C. Chang, J. Ning, and Y. Huang, "Public auditing for shared cloud data with efficient and secure group management," *Information Sciences*, vol. 472, pp. 107–125, 2019.

[36] A. D. Caro and V. Iovino, "jPBC: Java pairing based cryptography," in *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011*, Kerkyra, Corfu, Greece, 2011.