Hindawi
Security and Communication Networks
Volume 2021, Article ID 9921209, 15 pages
https://doi.org/10.1155/2021/9921209

Research Article

Blockchain-Based Cloud Data Integrity Verification Scheme with High Efficiency

Gaopeng Xie,[1] Yuling Liu,[1] Guojiang Xin ,[2] and Qiuwei Yang[1]

[1]*College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China*
[2]*School of Informatics, Hunan University of Chinese Medicine, Changsha 410208, China*

Correspondence should be addressed to Guojiang Xin; lovesin_guojiang@126.com

Received 13 March 2021; Revised 7 April 2021; Accepted 16 April 2021; Published 29 April 2021

Academic Editor: Zhili Zhou

With the large-scale application of cloud storage, how to ensure cloud data integrity has become an important issue. Although many methods have been proposed, they still have their limitations. This paper improves some defects of the previous methods and proposes an efficient cloud data integrity verification scheme based on blockchain. In this paper, we proposed a lattice signature algorithm to resist quantum computing and introduced cuckoo filter to simplify the computational overhead of the user verification phase. Finally, the decentralized blockchain network is introduced to replace traditional centralized audit to publicize and authenticate the verification results, which improves the transparency and the security of this scheme. Security analysis shows that our scheme can resist malicious attacks and experimental results show that our scheme has high efficiency, especially in the user verification phase.

1. Introduction

With lots of applications deployed in the cloud, user data are also collected centrally and handed over to the cloud. Cloud computing provides users with shared computing resources and storage resources and has multiple deployment models like private cloud, community cloud, public cloud, and hybrid cloud [1]. For users, storing data in the cloud can bring many benefits, such as reducing hardware investment costs, reducing the local storage burden, and supporting remote access. However, while cloud storage brings convenience, it also brings corresponding challenges. Highly centralized data and complex computing environments make user data subject to multiple threats [2]. Compared to traditional data centers, the cloud has more complex systems. The numerous components make the cloud more likely to be attacked. As the complexity of the systems increases, so do the vulnerabilities of the systems. Secondly, multitenants share cloud computing resources, making data more vulnerable to be damaged. In the cloud computing environment, the customized resources between tenants are usually isolated by adopting a logical method. A malicious attacker may pretend to be a tenant to launch an internal attack, violating other users' data. Finally, cloud service providers may deliberately hide that user data are corrupted or store data that users rarely access offline. Considering the large-scale outsourcing data and limited computing power, verifying outsourcing data integrity has become a vital issue in cloud storage.

The root problem of cloud data security lies in the trust between the cloud service provider (CSP) and the user. Failure of cloud devices, external attacks, or even the snooping of user data by the CSPs may result in leakage, loss, and damage of user data. On the other hand, even if user data is destroyed, users may not achieve effective accountability, and the CSP will evade responsibility and not recognize it. Therefore, the essence of the problem lies in the lack of trust between the two sides. Once problems occur, it is difficult for the challenged party to provide the basis agreed by both sides.

The traditional solution is to introduce a third-party auditor (TPA) to form a three-party authentication model [3], as shown in Figure 1. But there are still problems in this way, which cannot guarantee that the TPA will not cooperate

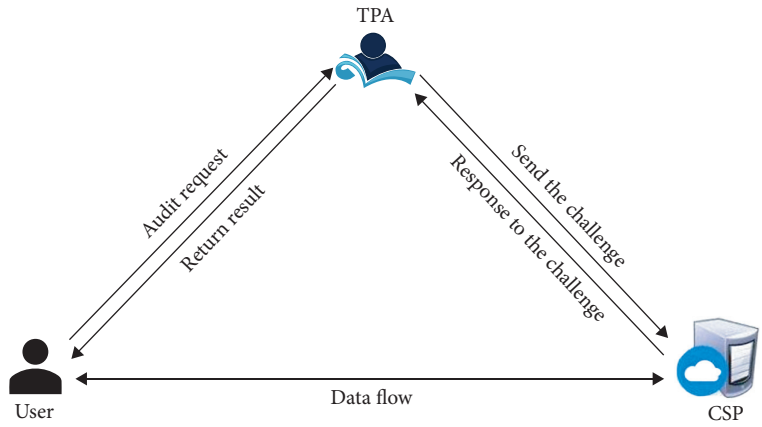

Figure 1: Traditional cloud storage model.

with the other party to cheat for interests or other reasons. However, the emergence of blockchain provides a new solution to this problem.

Blockchain is a type of chain structure that combines data blocks in chronological order, and it is a tamper-proof and nonforgeable distributed ledger guaranteed by cryptography [4]. All blockchain participants maintain the blockchain's node information, so all information on the blockchain is open and transparent. Once the information is released, it is permanently retained and cannot be tampered with. The open verification and tamper-proof features of the blockchain enable it to act as a trusted third party to address users' concerns in the cloud computing environment, and all results can be published to the blockchain for authentication and maintained by all users of the blockchain. Therefore, integrating the blockchain into cloud computing, using the blockchain's advantages to solve the disadvantages of the cloud computing environment, can more effectively provide users with data security.

With quantum computing development, traditional cryptography security is threatened, so a high-security signature scheme that can resist quantum attacks is necessary. This paper proposes a new cloud data integrity verification scheme based on blockchain and lattice signature. Our key contributions in this paper can be summarized as follows.

This paper proposes a protocol for cloud data integrity verification based on lattice signature. Under the small integer solution (SIS) assumption, the algorithm can resist quantum attacks and malicious attacks under the random oracle model and protect user data privacy. The user's computational complexity in the verification phase is greatly simplified by introducing the cuckoo filter, and the algorithm's efficiency is further improved.

This paper introduces a decentralized blockchain-based data integrity service to replace the traditional centralized data integrity service; this avoids the problem of untrustworthy TPA and further improves the reliability of the service.

This paper evaluates the correctness and feasibility of the proposed method through experiments. Experimental results show that the method is effective.

The rest of the paper is organized as follows. Section 2 provides an overview of the related work. Section 3 introduces the related background knowledge. Section 4 describes the proposed scheme in detail. Section 5 provides the security analysis. Section 6 presents the experiment and analyzes the performances of the scheme. Section 7 concludes the paper.

2. Related Work

With the rapid development of the Internet, there are more and more new hot areas, such as 5G [5], Internet of Things (IoT) [6], SDN [7], and cloud computing, which have spawned a lot of related research. In the field of cloud computing, there are some researches on the verification of data integrity, as well as data retrieval [8], image retrieval [9, 10], and so on. This section mainly introduces some related work on the cloud data integrity verification, lattice signature, and blockchain involved in this paper.

2.1. Cloud Data Integrity Verification. With the popularization of cloud storage, many scholars pay attention to remote data integrity audit. Cloud storage integrity verification refers to how users verify the integrity and availability of data stored in the cloud. To solve the security risks brought by cloud storage, many solutions have been proposed to ensure data integrity. At present, data integrity verification schemes can be divided into two categories: Provable Data Possession (PDP) schemes and Proofs of Retrievability (POR) schemes. The PDP schemes use the "challenge-response" mechanism to verify whether the user's data is correctly stored in the cloud; the POR schemes can recover some lost or damaged data. Besides, referring to the verifier's identity, the verification scheme can be divided into private verification scheme and public verification scheme. Compared with private verification, public verification with TPA better supports public audit, dynamic update, and efficient verification.

Ateniese et al. [3] proposed a PDP scheme for the first time in consideration of public audit based on the challenge-response mode, using RSA algorithm to generate key pairs

and using trusted third-party audit to verify data integrity instead of users. The CSP only needs to return the label information of the file block to verify the integrity of the file, reducing the computational and communication overhead of the data verification process. However, this scheme does not support the dynamic update operation of data. In order to support dynamic auditing of data, Ateniese et al. [11] proposed an extensible PDP. This scheme first calculates a limited number of verification tokens, and each token is linked to some data blocks, thereby ensuring that the scheme can support the modification of data in a prescribed manner, but the number of verifications and data updates performed by the verifier is limited, and only additional operations are supported. Each update requires recreating the remaining verification tokens. In this solution, the time complexity of the computational overhead of the CSP and the data owner is $O(t)$, and the time complexity of the communication overhead is $O(1)$. After that, a series of improvements have been proposed.

Erway et al. [12] proposed a new hierarchical authentication skip list data structure and used RASL to construct a PDP scheme that supports full data update but does not provide data privacy protection. Wang et al. [13] used a random mask to optimize the above scheme to protect users' privacy information, but the scheme is degraded and cannot support data block insertion. In addition, Zhou et al. [14] pointed out that the scheme could not resist the forgery attack; that is, the adversary with evidence could forge new legal evidence by repeatedly using a secret value to pass the integrity verification scheme.

Zhu et al. [15, 16] proposed a new index hash table (IHT) based data structure that supports dynamic operations and public audit. IHT can reduce the storage overhead of verification information. Yang et al. [17] used the properties of cryptography and bilinear pairing instead of mask technology to provide data privacy protection for data integrity schemes and support dynamic and batch auditing of data, but Ni et al. [18] proved that this method is fragile, because it does not provide response authentication and is vulnerable to enemy attacks.

Xu et al. [19] proposed an algorithm for detecting data verification results to resist counterfeit fraud attacks from untrusted verification results. The algorithm performs cross-validation by establishing a dual evidence mode of integrity verification proof and incredible check proof. Integrity verification proof is used to check the integrity of the data, and incredible check proof is used to determine the correctness of the data verification results, but the introduction of secondary verification evidence to cross-check verification results increases the computation and storage overhead. Shen et al. [20] proposed a new data integrity verification scheme that enables files in cloud storage to be shared securely without affecting privacy and integrity verification. Li et al. [21] proposed a provable data integrity method, which improves the verification efficiency by reducing the user cost during the initialization phase. Zhu et al. [22] proposed an integrity verification scheme based on a short signature algorithm (ZSS signature) for the IoT environment, which proved to be secure and efficient.

In the public audit verification model, most of the work assumes that the TPA is trusted to complete the entire data integrity verification work. However, in practical applications, the internal working process of TPA and its credibility need further research and proof. The techniques such as encrypted data blocks and random masks can prevent the original data from being leaked to the TPA. However, for users, TPA's internal structure and operation process are unknown. In reality, the TPA may collude with the CSP to attack the user or collude with the user or other CSP competitors to attack the CSP and return the wrong verification results.

In order to solve the malicious TPA, some methods have been proposed. The scheme of Huang et al. [23] used multiple TPAs for audit authorization and introduced the receive server and time server to ensure that the TPA completed the audit task accurately and timely. However, the credibility of the introduced server entity could not be confirmed. Wu et al. [24] proposed dividing the verification of TPA into complex calculation process and simple verification process. The former is handled by TPA, while the latter is verified by the user himself. Xiao et al. [25] used trusted hardware deployed on the cloud storage server to generate and store audit logs. Due to the limited performance of the hardware and the risks of deployment in untrusted cloud storage providers, its security and credibility need to be further verified. The above solutions do not really solve the task that TPA may not faithfully complete the audit requirements of cloud users.

2.2. Lattice Signature-Based Work. In recent years, lattice cryptographic schemes have been greatly developed in cryptography theory and achieved a series of research results. One of the major public problems of lattice signature schemes is to reduce the size of the verification key while implementing short signatures. In 2010, Boyen [26] proposed the first secure short signature scheme based on provable lattice under the standard model. Then Micciancio and Peikert [27] improved the hash function of Boven's scheme and further improved the security of the scheme. Subsequent provably secure digital signature schemes [28, 29] have made improvements in both the verification key size and the signature size but only realized existential unforgeability against chosen message attacks. Yang et al. [30] used the special algebraic structure of the ideal lattice to construct an identity-based signature scheme, which improved the efficiency of the scheme and simplified the signature. Under selected identity and fixed selection messages, the scheme satisfies unforgeability, but the scheme cannot achieve the anonymity if a single identity is signed. Later, Zhang et al. [31] not only designed a lattice signature scheme that can prove secure under the standard model with only $O(\log n)$ matrices for verification keys but also achieved existential unforgeability against chosen message attacks, that is, a completely secure lattice signature scheme, but the security of the scheme is limited by the number of message queries.

With the development of quantum computing, the security of traditional cryptography is threatened, so the research of lattice cryptography that can resist quantum computing has gradually expanded to various fields. Tian et al. [32] first applied lattice signature to partially blind signature field, saving a final round communication while confirming the validity of the signature. In the field of identity-based signature (IBS), Wang et al. [33] first used lattice signature to build an adaptive-ID secure IBS scheme with high space efficiency. Gao et al. [34] used lattice signature to ensure the security of the blockchain in the postquantum era and built an efficient cryptocurrency scheme based on lattice.

Due to the rapid development of quantum computing technology, existing public-key cryptographic standards will no longer be secure under quantum computing. As a representative of the resistance to quantum computing attacks, more and more scholars begin to devote to the study of lattice signature.

2.3. Application of Blockchain. Because of the malicious TPA, schemes based on cryptographic principles instead of credit are needed, so that both the user and the CSP can reach an agreement to objectively and honestly verify the data integrity. The birth of blockchain provides a guarantee for "trust." It uses a consensus algorithm to ensure data consistency between nodes and an encryption algorithm to ensure data security.

Hardjono et al. [35] proposed a privacy protection method for debugging IoT devices into the cloud ecosystem based on the blockchain, enabling anonymous registration of the device, and being able to prove its manufacturing source. Heilman et al. [36] combined smart contracts with blind signature technology to achieve transaction anonymity. Liang et al. [37] designed a system called ProvChain, which can collect the provenance data, store and verify it through blockchain, and provide a complete provenance data record when needed. Zhu et al. [38] developed a blockchain-based file management system to specifically address the problem that project documents are vulnerably tempered with. In the field of cloud data integrity verification, Yue et al. [39] proposed a blockchain-based verification scheme in P2P cloud storage and verified data by using Merkle trees. Wang et al. [40] deeply combined the blockchain with the PDP scheme to create the first blockchain-based PDP model with high efficiency and security. To against procrastinating auditors, Zhang et al. [41] proposed a certificateless verification scheme with blockchain technology, but this scheme can only avoid procrastinating auditors and cannot handle other issues such as TPA colluding with others. Tian and Li [42] proposed another cloud verification scheme, which is a cloud federation of TPA based on blockchain to solve the unreliability of TPA. The decentralization of the blockchain is very compatible with the IoT, so there are studies using blockchain to verify the integrity of the data of IoT devices. Liu et al. [43] proposed a blockchain-based framework to achieve decentralized data integrity verification of IoT data stored in a semitrusted cloud. In addition, Wei et al. [44] proposed an integrated model using blockchain technology. They use mobile agent technology to deploy the distributed virtual machine agent model in the cloud. It can ensure the reliability of data storage, monitoring, and verification, which is also essential for building a blockchain integrity protection mechanism.

The characteristics of openness, transparency, and data traceability of the blockchain make it play an essential role in finance, medical care, and the IoT. In cloud data integrity verification, the combination of integrity verification and blockchain is still in the exploration stage. Therefore, in this paper, we propose a new blockchain-based cloud data integrity verification method, using blockchain to solve the untrustworthy problem of TPA.

3. Preliminaries

This section will introduce some background related to our scheme, which is mainly divided into three parts: the background related to the lattice signature algorithm, the background of blockchain, and the background of the cuckoo filter. The lattice signature background mainly includes the introduction of lattice and the difficult problem on the lattice. The security of the lattice signature algorithm in our scheme is based on the SIS problem, so we introduce the SIS problem; what is more, to eliminate the situation that the distribution of the signature results related to the user's private key distribution, we also need to use rejection sampling theorem.

3.1. Lattice

Definition 1. Let $B = (b_1, \ldots, b_m) \subset R^n$ be a set of linearly independent vectors, then the lattice Λ generated by B is $\Lambda = \left\{Bc = \sum_{i=1}^{M} b_i c_i | c_i \in Z, 1 \le m \le n\right\}$, and B is called the basis of the lattice Λ. The length of the basis $\|B\|$ is the length of the longest vector in B. The standard orthogonal basis of B obtained by the standard Schmidt orthogonalization is $\widetilde{B} = (\widetilde{b_1}, \ldots, \widetilde{b_n})$.

Definition 2. Let q be a prime number, $A \in Z_q^{n \times m}$, and $u \in Z_q^n$, and define

$$\begin{aligned} \Lambda_q^{\perp}(A) &\Lambda \{e \in Z^m \text{ s.t. } Ae = 0 \,(\mathrm{mod}\, q)\}, \\ \Lambda_q^{u}(A) &\Lambda \{e \in Z^m \text{ s.t. } Ae = u \,(\mathrm{mod}\, q)\}. \end{aligned} \tag{1}$$

Lemma 1. *Let $q \ge 3$ be an odd number and $m = 5n \log q$; then there is a probabilistic polynomial time algorithm $TrapGen(q, n)$, output matrix $A \in Z_q^{n \times m}$, and $T \in Z_q^{m \times m}$, where A is nearly uniform on $Z_q^{n \times m}$, T is a basis of lattice $\Lambda_q^{\perp}(A)$ and satisfies $\widetilde{T} \le O(\sqrt{n \log q})$, and $T \le O(n \log q)$ except for a negligible probability.*

3.2. SIS Problem

Definition 3. SIS problem. Given an integer q, a matrix $A \in Z_q^{n \times m}$, and a real number β, find a set of nonzero

solutions such that the homogeneous equation $Ae = 0\, mo\, d\, q$ holds, where e satisfies that $e_2 \le \beta$.

SIS assumption. For some bounded polynomial functions $q(n)$, $m(n)$, $\beta(n)$ with n as the independent variable, if the probability of enemy A successfully solving the SIS problem in any polynomial time is negligible in the case of unknown trap door, then the SIS assumption is valid.

3.3. Discrete Gaussian Distribution. The Gaussian distribution with standard deviation $\sigma \in R$ and center $v \in R$ is defined as $\rho_{v,\sigma}(x) = \exp((-(x-v)^2)/2\sigma^2)$. For $x, v \in R^n$, $\rho_{v,\sigma}(x) = \exp((-x-v^2)/2\sigma^2)$. Discrete Gaussian distribution on Z^m, centered on $v \in Z^m$, standard deviation $\sigma \in R$, defined as $D^m_{v,\sigma}(x) = ((\rho_{v,\sigma}(x))/(\rho_\sigma(Z)^m))$. If the discrete Gaussian distribution centered at 0 over Z^m, the expression can be simplified to $D^m_\sigma(x) = ((\rho_\sigma(x))/(\rho_\sigma(Z)^m))$.

Lemma 2. *If* $k > 1$, $\Pr[z > k\sigma\sqrt{m}; z \in D^m_\sigma] < k^m e^{(m/2)(1-k^2)}$. *Further, for any vector* $v \in R^m$ *and* $\sigma, r > 0$, $\Pr[|z, v| > r; z \in D^m_\sigma] \le 2e^{-(r^2/(2v^2\sigma^2))}$.

Lemma 3. *For any* $v \in Z^m$, *if* $\sigma = \alpha v$, *where* $\alpha > 0$, $\Pr[(D^m_\sigma(z)/D^m_{v,\sigma}(z)) < e^{(12/\alpha+1/2\alpha^2)}; z \in D^m_\sigma] = 1 - 2^{-100}$.

4. Rejection Sampling Theorem

If f and g are probabilistic distribution functions, there exists $M \in R$, and $f(x) \le M \cdot g(x)$ is satisfied for any x. If the sampled points z are taken from the distribution g and a pair (z, c) is output with probability $P_r = (f(z)/(Mg(z)))$, then (z, c) can be regarded as an instance of rejection sampling, the distribution of the output result is f, and the number of times required to output an instance is M.

Some basic properties of rejection sampling theorem:

(1) $\Pr[z > \omega(\sigma\sqrt{\log m}); \ z \in_R D^1_\sigma] = 2^{-\omega\sqrt{\log m}}$; $\Pr[z > 12\sigma; \ z \in_R D^1_\sigma] < 2^{-100}$.

(2) $\forall z \in Z^m, \sigma \ge \sqrt{\log(3m)}; \ D^m_{v,\sigma} \le 2^{-m+1}$.

(3) $\Pr[z > 2\sigma\sqrt{m}; z \in_R D^m_\sigma] < 2^{-m}$.

4.1. Blockchain. Blockchain is an important concept of Bitcoin [45], which is essentially a decentralized database and at the same time serves as the underlying technology of Bitcoin. A blockchain is a series of data blocks generated by using cryptographic methods, as shown in Figure 2. Each block contains information about a Bitcoin network transaction used to verify its data's validity and generate the next block.

In a narrow sense, blockchain is a chained data structure that sequentially combines data blocks according to the time sequence, and a distributed ledger that cannot be tampered with or forged through cryptography.

In a broad sense, blockchain technology uses blockchain data structures to verify and store data, uses distributed node consensus algorithms to generate and update data, uses cryptography to ensure data transmission and access, and uses smart contracts composed of automated script code to program and manipulate data.

Blockchain has the characteristics of decentralization, immutability, traceability, collective maintenance, openness, and transparency. These characteristics ensure the blockchain's honesty and transparency and lay the blockchain's foundation to create trust. The blockchain's rich application scenarios are based on the fact that the blockchain can solve information asymmetry and achieve collaborative trust and concerted action among multiple subjects.

5. Cuckoo Filter

Cuckoo filter [46] is a random data structure with a simple structure and high space efficiency. Compared with the bloom filter, the cuckoo filter has the advantages of good query performance, high space utilization efficiency, and support for reverse operation. It provides two possible storage locations for each keyword, dynamically relocates existing keywords, makes room for new keywords during insert operations, and quickly locates keywords during lookup operations. The cuckoo filter's expected insertion time complexity is still $O(1)$, although repeated relocations are required. The insertion process is shown in Figure 3.

We can calculate the two candidate buckets for a keyword through the following formula:

$$h_1(x) = \text{hash}(x), \tag{2}$$

$$h_2(x) = h_1(x) \oplus \text{hash}(x's \text{ fingerprint}). \tag{3}$$

Cuckoo filter only store fingerprint values instead of original values, so equation (2) can ensure that $h_1(x)$ can also be calculated through $h_2(x)$ and the fingerprint. So, once you know the current bucket k and its fingerprint, you can calculate another bucket by

$$k' = k \oplus \text{hash}(\text{fingerprint}). \tag{4}$$

There are three cases when inserting: the first case is that both buckets are empty, and then a vacant position is randomly selected to insert the item. The second case is that only one bucket has a vacant position so directly inserted the item into this position. The third case is that neither bucket is empty; then randomly choose a bucket, swap the item in the bucket with the item to be inserted, and then relocate the kicked item by equation (3); if the relocated bucket is still not empty, then continue to kick out the original item and relocate it, and repeat this process until all elements are inserted.

The cuckoo filter also supports lookup and delete operations. The lookup operation only needs to query whether the item is in one of the corresponding two buckets. The delete operation is similar; remove the item from the corresponding bucket. The detailed description of these operations can be found in the article [46].

The fingerprint-based insert algorithm enables the insert operation to use only the bucket's information, without reretrieving keywords. Through equations (2) and (3), the dynamically adding and deleting elements can be realized.

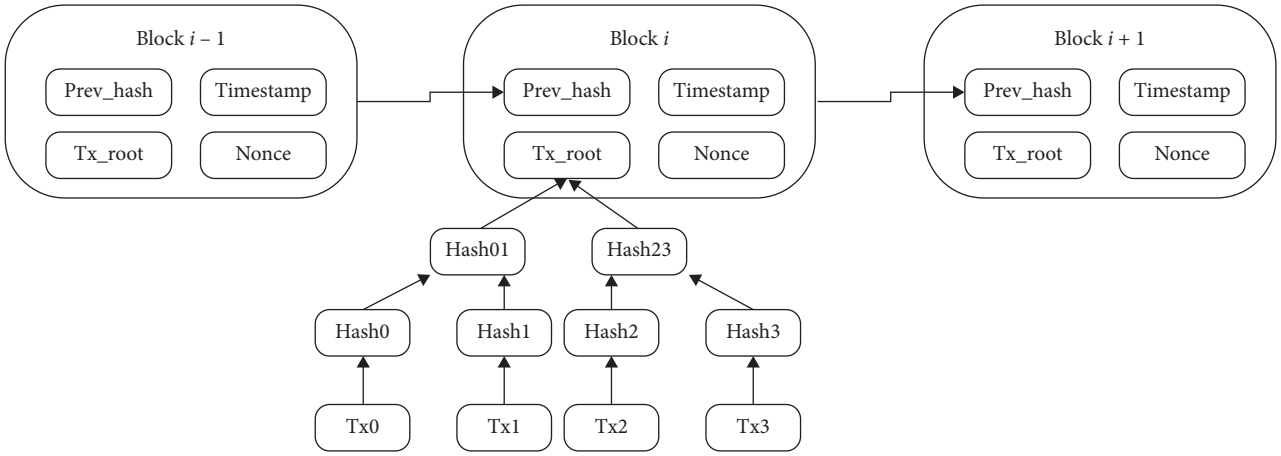

Figure 2: The basic structure of blockchain.

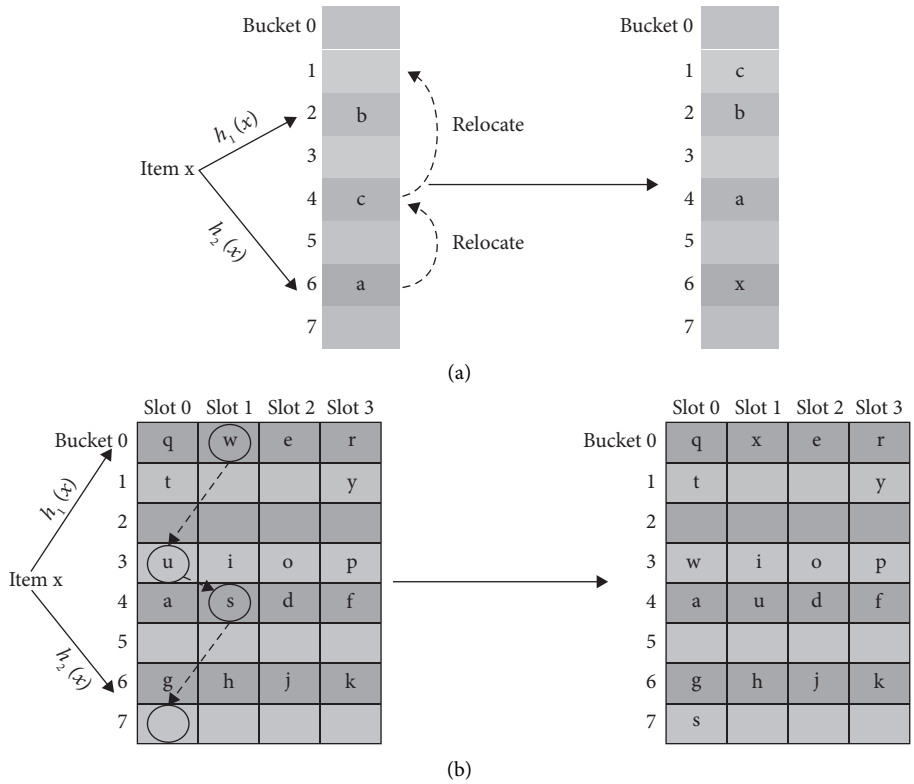

Figure 3: Illustration of cuckoo filter. (a) Each bucket with one slot. (b) Each bucket with four slots.

The cuckoo filter application, which has the advantages of efficient computation and storage, can reduce the storage and calculation overhead of the verification process to the data integrity verification.

6. The Proposed Data Integrity Verification Scheme

6.1. Design Goals. In order to complete the data integrity verification safely and efficiently, our proposed data integrity verification scheme should have the following properties:

(1) Dynamic integrity verification: users may often need to update the data uploaded to the CSP. The proposed scheme needs to support dynamic changes of data, including support for data insertion, data deletion, and data modification.

(2) Resist quantum attack: with the advent of the quantum era, lattice cryptography plays an increasingly important role in the fields of cryptography and information security. The proposed scheme should be able to resist the attack of quantum computer and be safe in the quantum environment.

(3) Trusted audit: when users upload data to the CSP, their control over the data is greatly reduced, and traditional data verification methods may not be applied to the cloud environment. At the same time, the CSP may have the problem of maliciously modifying the results of data integrity verification. Therefore, the proposed data integrity verification scheme must ensure that the results of data integrity verification are fair and credible.

6.2. System Model. Most integrity verification protocols use TPA to communicate the interaction between the user and the CSP, improving data integrity verification efficiency and reducing the user's computing and storage overhead. However, since TPA performs the verification, the reliability of TPA is in doubt, and there are potential threats such as conspiring with CSP to deceive users or fake proof. An ideal public audit institution should have the following characteristics: no additional computing and storage costs, no data privacy disclosure, and most importantly, fairness and justice. Therefore, we introduce blockchain as the third-party audit to replace traditional centralized audit. The system is mainly divided into three types of participants.

6.2.1. User. The user has the ownership of the data files and the local storage space is limited, so user chooses to entrust the files to the CSP. For the sake of cloud data security, the user will check the integrity of the uploaded data from time to time.

6.2.2. CSP. The CSP has a large storage space and strong computing capabilities. It makes money by providing storage and computing services for various users, enabling users to upload and download data anytime and anywhere. But the CSP is only responsible for storing the data and does not ensure data security.

6.2.3. Blockchain. A third-party audit platform between the user and the CSP is responsible for forwarding and recording the user and the CSP interactions during the data integrity verification process. When users dispute with CSP, the blockchain's records can be submitted to the arbitration institution as valid evidence. All participants jointly maintain the blockchain network, and the behavior of users and CSP is jointly monitored to ensure the system's normal operation.

6.3. Scheme Details. The proposed scheme uses the lattice signature algorithm to sign the files on the user side, the cuckoo filter is also used to simplify the user verification process, and the blockchain network is introduced to record the interaction between the user and the CSP. The scheme mainly includes six parts: KeyGen(), SigGen(), Upload(), Challenge(), ProofGen(), and Verify(). The process is shown in Figure 4 and the details are given below.

KeyGen(): This phase is performed by the user to generate the user's public key and private key. First, prepare the hash function H distributed on B_k^n as the random oracle model, where B_k^n is the set of binary vectors with length n and weight k. Then generate a random matrix $S \in Z_{2q}^{m\times n}$ as the user's private key and a matrix $A \in Z_{2q}^{n\times m}$ as the user's public key and the matrix A needs to satisfy $AS = A(-S) = qI_n \pmod{2q}$, I_n represents the n-dimensional identity matrix. The way to generate the key pair is easy to implement and the whole process is efficient.

SigGen(): performed by the user. On one hand, the file to be uploaded is divided into data blocks to construct the signature set; on the other hand, Merkle hash tree (MHT) and cuckoo filter are constructed based on the signature set. The signature process can be specifically divided into the following steps.

Step 1: The user divides the file equally into blocks of the same size $F = \{F_1, \ldots, F_n\}$.

Step 2: The user uses random function to generate a secret value τ and blinds the file blocks with τ, $\mu_i = F_i + f_\tau(i, ID_F)$, where ID_F is the identification of the file and μ is the blinded data.

Step 3: The user randomly samples vector y_1 from the discrete Gaussian distribution D_σ^m and vector y_2 from D_σ^n; set $u = Ay_1 + y_2$.

Step 4: Calculate $c_i = H(Au \bmod 2q, \mu_i)$, where A is the public key and μ is the message to be signed; calculate $z_i = u + (-1)^b Sc_i$, where b is an element randomly sampled from the set $\{0, 1\}$.

Step 5: The signature pair (z_i, c_i) is output by rejection sampling theorem with probability $(1/(M \exp(-||Sc||^2/2\sigma^2)\cosh(\langle z, Sc\rangle/\sigma^2)))$. In particular, if the rejection sampling has no output, then the signature process will restart from step 2.

Step 6: After the signature pair (z_i, c_i) is output, the user will calculate z_i. If $z_i > B_2$ or $z_{i\infty} \geq (q/4)$, then reject and restart the signature process. Else verify whether $c_i = H(Az_i + qc_i \bmod 2q, \mu)$; if equal, the signature generation completes.

The signature algorithm adopts the rejection sampling theorem to make the distribution of the signature (z, c) independent of the private key S and increase the security of the signature. Then the user needs to build MHT with the signature set $\Phi = \{c_i\}, 1 \leq i \leq n$ as the leaf nodes. Cuckoo filter is also created based on the leaf nodes of the MHT. The specific steps to construct the cuckoo filter are as follows. First, create an empty hash table. Then the two buckets corresponding to each MHT leaf node according to equations (2) and (4) are calculated. After all the nodes have been inserted according to the insertion algorithm, the cuckoo filter is built.

Upload(): performed by the user and the CSP. After completing the above operations, the user constructs an upload request $\{\text{upload}, ID_{\text{user}}, ID_{\text{CSP}}, ts, F, H(F), ID_F, \Phi, \sigma_{\text{user}} = \text{sign}(H(R))\}$, sends to the smart contract, and then forwards to the CSP, where ts is a timestamp and

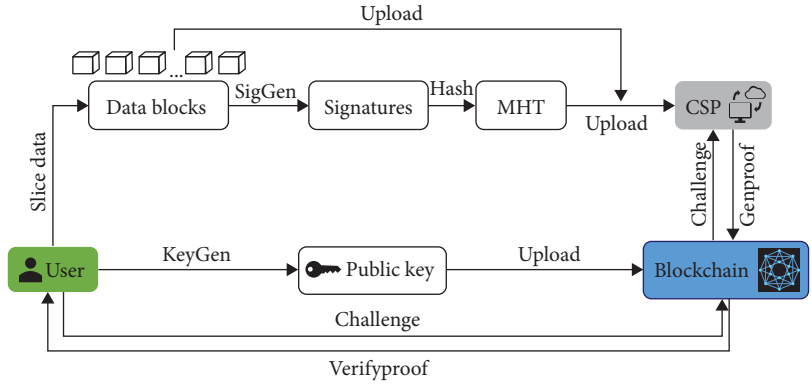

Figure 4: Verification process of the system.

$\text{sign}(H(R))$ is the signature generated by the private key on the root node of MHT. Upon receiving the upload request from the user, the CSP first verifies σ_{user} and calculates $H'(F)$ through F to verify whether $H'(F) = H(F)$. If equal, then it retains F and sends a response $\{\text{upload}, ID_{\text{CSP}}, ID_{\text{user}}, ts, ID_F, 1, \sigma_{\text{CSP}}\}$ to the user through smart contract, where $\sigma_{\text{CSP}} = \text{sign}(ID_{\text{CSP}} \| ID_{\text{user}} \| ts \| ID_F \| 1)$, 1 means success and 0 means failure. The user verifies σ_{CSP}; if passed, the user can delete the local file and the file upload process is complete. Otherwise, the file upload fails and the user restarts the upload project.

Challenge(): Performed by the user, the user sends the challenge request to the CSP through the smart contract. When the user wants to verify the data integrity, a x-element subset $I = \{s_1, \ldots, s_x\}$ from set [1, n] is randomly selected as the audit request and then a challenge set chal = $\{i\ d, I\}$ is generated. The user constructs an audit request $\{\text{audit}, ID_{\text{user}}, ID_{\text{CSP}}, ts, ID_F, \text{chal}, \sigma_{\text{user}}\}$ and sends to the CSP, where $\sigma_{\text{user}} = \text{sign}(ID_{\text{user}} \| ID_{\text{CSP}} \| ts \| ID_F \| \text{chal})$.

ProofGen(): Performed by the CSP. After the CSP receives the user's audit request, it first verifies the user's signature. If valid, the CSP locates the location of each file block, then calculates the corresponding signature $\Phi' = \{c_i'\}, 1 \le i \le x$ with the public key, then constructs the proof $\{ID_{\text{CSP}}, ID_{\text{user}}, ts, ID_F, F, \Phi', \sigma_{\text{CSP}}\}$, and sends to the user, where $\sigma_{\text{CSP}} = \text{sign}(ID_{\text{user}} \| ID_{\text{CSP}} \| ts \| ID_F \| \Phi')$.

Verify(): After the user receives the proof returned by the CSP, the user first verifies the validity of the signature. Then the lookup operation of cuckoo filter is performed to check whether all signatures exist in the cuckoo filter. If all signatures exist in the cuckoo filter, the data integrity verification is passed; otherwise, the data are compromised.

6.4. Dynamic Operation. Generally speaking, files are not immutable after being uploaded by users. In practical applications, users will need to update files, such as add, delete, and modify. So, we use MHT to support dynamic operation. Simultaneously, the proposed scheme reduces the complexity of the verification process by introducing cuckoo filter, so the dynamic operation of the scheme involves two parts. The first part is the update operation of MHT, and the second part is the update operation of the cuckoo filter.

The specific steps to update the file are as follows.

Step 1: The user first calculates the signature pair of the updated file (z_i^*, c_i^*) and generates the corresponding update request Update = $((M/I/D), i, F_i^*, c_i^*)$, where M represents the modify operation, I represents the insert operation, D represents the delete operation, and F_i^* represents the updated file.

Step 2: The user constructs the update request $\{\text{Update}, ID_{\text{user}}, ID_{\text{CSP}}, ts, \sigma_{\text{user}}\}$ to the CSP, where $\sigma_{\text{user}} = \text{sign}(ID_{\text{user}} \| ID_{\text{CSP}} \| ts \| \text{Update})$. After receiving the request, CSP updates the data on the cloud according to the request. If Update = (M, i, F_i^*, c_i^*), the CSP replaces F_i with F_i^* on the MHT. If Update = (I, i, F_i^*, c_i^*), the CSP inserts a new node F_i^* after the leaf node F_i and updates the MHT, as shown in Figure 5. If Update = (D, i, F_i^*, c_i^*), the CSP deletes the leaf node F_i and updates the MHT, as shown in Figure 6. After the file is updated, the CSP gets a new root node R^* of MHT and returns the update proof $\text{Proof}_U = \{ID_{\text{CSP}}, ID_{\text{user}}, ID_F, i, \Phi, H(F_i), R^*, \sigma_{\text{CSP}} = \text{sign}(H(R))\}$ to the user.

Step 3: After receiving the proof, the user first verifies the signature $\sigma_{\text{CSP}} = \text{sign}(H(R))$ and then uses $\{ID_F, i, \Phi, H(F_i), H(F_i^*)\}$ to calculate whether the generated root node R' is the same as R^*. If equal, the update operation is successful, and the user calculates the signature $\text{sign}(H(R'))$ for the new root node of MHT and sends response $\{\text{Update}, ID_{\text{user}}, ID_{\text{CSP}}, ts, 1, \sigma_{\text{user}} = \text{sign}(H(R'))\}$ to the CSP.

Step 4: The user deletes the deleted files from cuckoo filter and inserts the updated nodes. After the cuckoo filter update is complete, the file update operation is complete, and the user deletes the local file.

6.5. Security Analysis. In 2016, NIST (National Institute of Standards and Technology) released "Report on post-quantum cryptography" [47]. According to the report, due to the rapid development of quantum computing technology, most existing public-key cryptographic standards will no longer be secure under quantum computing. Up to now, there is no effective polynomial time algorithm that can solve the difficult problem based on lattice, so the cryptosystem based on lattice can

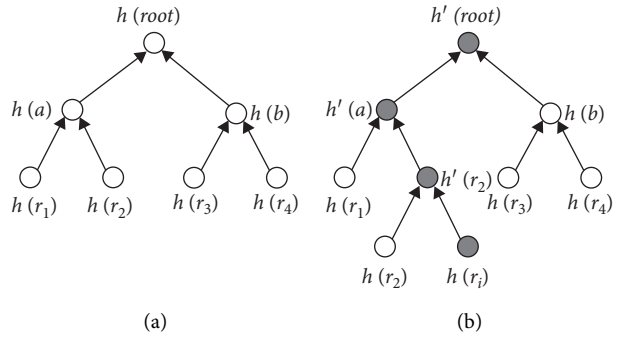

Figure 5: Insert operation of MHT.

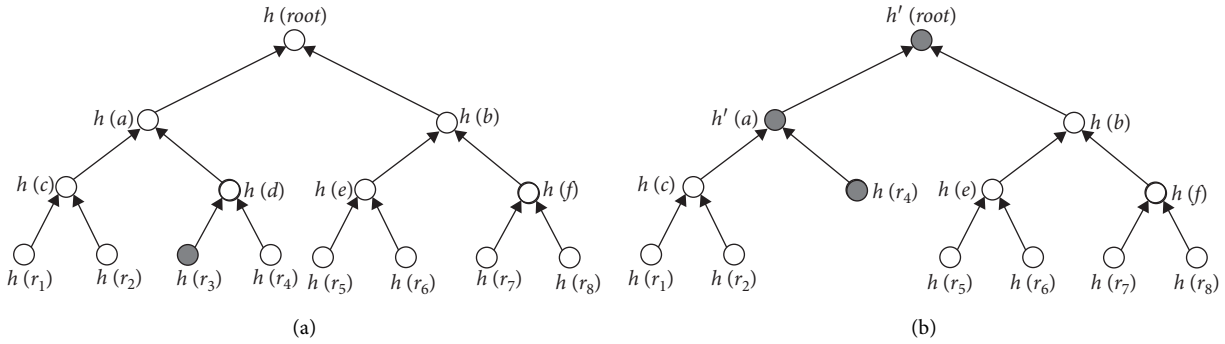

Figure 6: Delete operation of MHT.

effectively resist quantum attacks. Therefore, our proposed scheme meets the security requirements of post-quantum cryptography. Then, we will analyze the security of the proposed scheme.

6.6. Privacy

Theorem 1. *In the entire integrity verification process, no matter who obtains the interaction information between the user and the CSP, the user's private information cannot be parsed.*

Proof. During the verification process, there are three main parts of the interaction between the user and the CSP. The user uploads $\{\mu_i, c_i, i, i\ d\}$ to the CSP, the user sends chal = $\{i\ d, I\}$ to the CSP, and the CSP returns proof to the user.

After the user divides the file into blocks, the data blocks can be blinded by random mask technology; that is, $\mu_i = F_i + f_\tau(i, \text{name})$. Thus, no one except the user can parse the original file data. In addition, the user only needs to upload $\Phi = \{c_i\}, 1 \le i \le n$ to the CSP, from the public key, and Φ cannot derive any valid information. The challenge information chal = $\{i\ d, I\}$ only contains the block number to be verified, the proof information returned by the CSP is similar to the information uploaded by the user, so valid information cannot be derived from it.

In summary, secret data will not be disclosed during the data integrity verification process, and user privacy is guaranteed. □

6.7. Correctness

Theorem 2. *Given the lattice signature pair (z, c) and the data file f, the verifier can check the correctness of the signature pair.*

Proof. Prove the correctness of the signature pair is equivalent to prove the correctness of the equation $H(Ay \bmod 2q, \mu) = H(Az + qc \bmod 2q, \mu)$. So, the correctness can be proved as follows:

$$\begin{aligned} Az + qc &= A\left(y + (-1)^b Sc\right) + qc = Ay + (-1)^b ASc + qc \\ &= Ay + (qI_n)c + qc = Ay \bmod 2q. \end{aligned} \tag{5}$$

Hence, $H(Ay \bmod 2q, \mu) = H(Az + qc \bmod 2q, \mu)$, and the correctness of the scheme can be proved. □

6.8. Unforgeability

Theorem 3. *Assume that there is a polynomial time algorithm F which makes up to s queries on the signature oracle and h queries on the random oracle H, and successfully forge the signature with a nonnegligible probability δ. Then there is a polynomial time algorithm that can solve $SIS_{q,n,m,\beta}$ ($\beta = 2B_2$) difficult problem.*

Proof. Suppose that $c' = H(Az' + qc' \bmod 2q, \mu\prime)$ is F's response to the signature query; then we can get that $c' =$

$H(Az + qc' \bmod 2q, \mu)$ hold. If $\mu \neq \mu\prime$ and $z \neq z'$, then the probability that the two hash values are equal is infinitely close to 0. Then we can find $\mu = \mu\prime$ and $z = z'$; thus, $A(z - z') = 0 \bmod 2q$. Since $z \neq z'$ and $z_\infty' \leq (q/4)$, so $z - z' \neq 0 \bmod q$ and $z - z' \leq 2B_2$. Thus, there is no such response that can forge the signature.

Another situation is that c' is F's response to the random oracle query, then we can get $c^* \leftarrow B_k^n$, and the probability that the forger uses $c' \neq c^*$ to forge the signature is $\xi = (\delta - (1/|B_k^n|)) \cdot ((\delta - (1/|B_k^n|)/t) - (1/|B_k^n|))$. That is, F uses the signed data μ to generate the signature pair (c', z') with the probability ξ, and $Az + qc' = Az' + qc^*$. Then $A(z - z') = q(c' - c^*) \bmod 2q$. Since $c' - c^* \neq 0 \bmod 2q$, so $z - z' \neq 0 \bmod 2q$. Since $z - z_\infty' < (q/2)$, so $z - z' \neq 0 \bmod q$. So, the condition cannot be met and the signature cannot be forged.

In summary, there is no probabilistic polynomial algorithm that can forge the signature; the proposed scheme can resist malicious attacks. □

7. Security Analysis of Blockchain Network

Blockchain is a decentralized and untrusted distributed shared ledger system that combines data blocks in a chronological order to form a specific data structure. It is cryptographically guaranteed to be tamper-proof and unforgeable. From a data perspective, blockchain is a distributed database that cannot be changed in practice. Traditional distributed databases only maintain data at a central server node, and other nodes store only data backups. The data storage on the blockchain is completely distributed; that is, all nodes jointly participate in data maintenance. The tampered or destroyed data of a single node will not affect the data stored in the blockchain. Only more than 51% of the nodes jointly launching an attack can change the data on the blockchain. Therefore, the data on the blockchain can be considered immutable to achieve secure storage of the data.

On the other hand, since the blockchain runs automatically, there is no problem with procrastinating auditors, and it is impossible to collude with CSP. Therefore, when users have disputes with CSP, the records on the blockchain can be used as valid evidence. Simultaneously, by replacing TPA with blockchain, users' sensitive private information is accessible only to themselves, which can avoid obtaining users' privacy information during the TPA verification process and guarantee the users' privacy security.

In conclusion, using blockchain as the third-party authentication platform can ensure availability, security, efficiency, and user privacy.

8. Experiment Results and Evaluation

8.1. Experiment. The experiment is running on the computer with the following settings:

CPU: Intel Core i7-8750H @ 2.20 GHz

Memory: 16 GB DDR4 2667 MHz

HD: 5400RPM Western Digital 1TB HDD

OS: Ubuntu 16.04 system.

The algorithm and cuckoo filter are conducted by C programming language with the pairing-based cryptography (PBC) library version 0.5.14, GNU multiple precision arithmetic (GMP) library version 6.2.0, and the Openssl version 1.0.2n. The security parameter of the lattice signature is set to $\lambda = 128$ bits. The size of the file to be signed is all 1 MB. All experimental data are the average result of 100 experiments. The blockchain network is conducted on Hyperledger Fabric. Hyperledger Fabric is a leading open source, general-purpose blockchain structure built for enterprises. Since Hyperledger does not require mining, it does not require strong hardware support, nor consume resources, and its allowable transactions per minute are much greater than Ethereum. The framework of the blockchain we implemented is shown in Figure 7. The user and the CSP communicate through the smart contract. The communication mode is that the user or the CSP first initiates the communication request to the smart contract, then the smart contract writes the request into the ledger and then broadcasts it to other nodes, and the communication is completed after the corresponding node receives.

9. Computational Overhead

To evaluate the computational overhead of our scheme, we denote the computational overhead required for the add operation as Add, the multiply operation as Mul, the hash operation as Hash, and the mod operation as Mod. During the signature generation process, the main calculation operations are $H(Au \bmod 2q, \mu)$ and $u + (-1)^b Sc$. Assume that the file is divided into n file blocks, then the client needs to generate n corresponding signatures. Besides, the rejection sampling theorem is used during the signature generation; it needs to repeat the signature generation process M times on average to output one signature. Hence, the computational cost of the client is $3nM\text{Mul} + 2nM\text{Add} + nM\text{Mod} + nM\text{Hash}$. Assume that the challenge request sent to the CSP contains c random numbers, then the computational cost of CSP is $2c\text{Mul} + c\text{Add} + c\text{Mod} + c\text{Hash}$.

9.1. Communication Overhead. As for the communication overhead, during the challenge phase, the audit request $\{i\ d, I\}$ contains only numbers representing the file number to be validated, so the communication cost in this phase is negligible. During the GenProof phase, the CSP needs to return the signatures and the relevant files requested by the user, so the communication cost is $nl_S + nf$, where n represents the number of blocks required in the challenge phase, l_S represents the signature length of each file, and f represents the size of the file block.

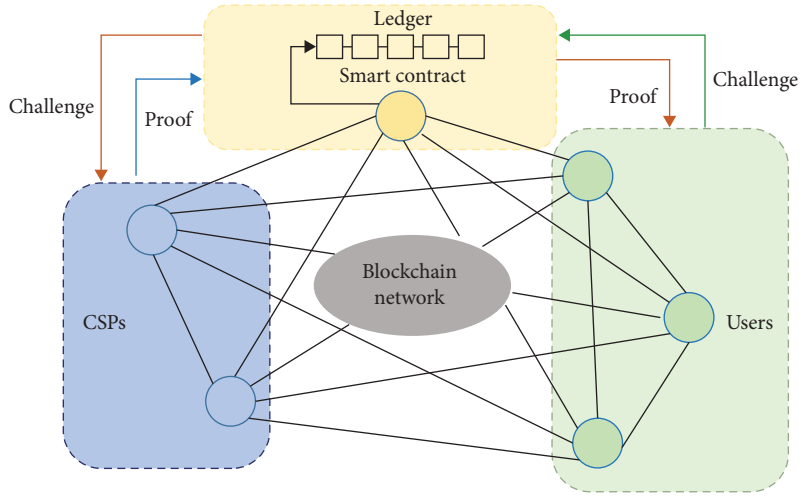

Figure 7: The framework of the blockchain network.

Table 1: Characteristics analysis of different cloud data integrity verification schemes.

Scheme	Dynamic operation	Privacy protection	Resistant to quantum attack	Trusted third party
[20]	No	Yes	No	No
[21]	Yes	Yes	No	No
[24]	Yes	No	No	No
[40]	No	Yes	No	Yes
Our scheme	Yes	Yes	Yes	Yes

9.2. Characteristic Analysis. Table 1 shows the analysis of the characteristics of different cloud data integrity verification schemes. The proposed scheme uses the blockchain network instead of traditional TPA, which solves the untrustworthy problem of TPA. CSP and blockchain only store files that can be disclosed, and only users can access private information. The lattice signature scheme is also proposed to enhance the signature security and meet the requirements of resisting quantum computing attacks. The scheme's efficiency analysis mainly analyzes the lattice signature scheme's efficiency and the efficiency of the user's integrity verification.

The characteristic analysis of different lattice signatures is shown in Table 2, where n is the security parameter, m is the dimension of the lattice, and μ is the length of the data block. From the result, we can see that, on the one hand, our proposed scheme avoids the use of expensive Gaussian sampling and improves efficiency. On the other hand, it uses a random oracle model to improve the security of the scheme. In terms of signature length, our scheme's signature length is shorter.

10. Performance Analysis

In this section, we will evaluate the performance of the proposed scheme through experiments. We evaluate the performance of the blockchain network first; Figure 8, respectively, shows the throughput and consensus time of the blockchain.

It can be seen from the results that the blockchain network has a throughput of thousands of transactions per second and a consensus time of milliseconds, which can fully meet the normal needs of the system. Then we evaluate the performance of the signature algorithm. Our method is compared with the BLS signature proposed by Wang et al. [13] and the ZSS signature proposed by Zhu et al. [22]. We conduct experimental comparisons from three aspects: signature generation cost, proof generation cost, and verification cost.

The signature generation time comparison of the three schemes is shown in Figure 9. The result shows that our scheme has a lower computational overhead than the other two schemes in terms of signature generation. Since the BLS signature and the ZSS signature use bilinear mapping, many exponential operations with high computational overhead are involved. Our method's main computational overhead is multiplication operation, hash operation, and repeated calculations in the signature process caused by the rejection sampling theorem. Although the ZSS signature is optimized in many aspects such as more efficient hash operations, our scheme is still more efficient.

Figure 10 shows the comparison result of the proof generation time. There is a linear relationship between the proof generation time and the number of challenging blocks. Like the signature generation process, the BLS signature needs to calculate an exponential operation, a hash operation, and a bilinear mapping operation. However, the computational overhead of the ZSS signature in the proof generation stage is higher than that of the BLS signature, since although the ZSS signature does not involve exponential calculations, the amount of data to be calculated is much larger, so it takes the most time. In our scheme, the proof generation stage only involves multiplication, hash,

Table 2: Comparisons of different lattice signature schemes.

Scheme	Public-key length	Private key length	Signature length	Whether sampled	Random oracle model
[27]	$(nm + (\|\mu\| + 2)n^2k + n)\log q$	$mnk \log q$	$m+2nk \log q$	Yes	No
[32]	$nk \log q$	$mk \log(1 + 2\ d)$	$2m \log(12\sigma) + 2k \log b$	No	Yes
[34]	$(mn + dm)\log q$	$m^2 \log q$	$m \log q$	Yes	No
Our scheme	$nm \log 2\ q$	$nm \log 2\ q$	$m \log(12\sigma)$	No	Yes

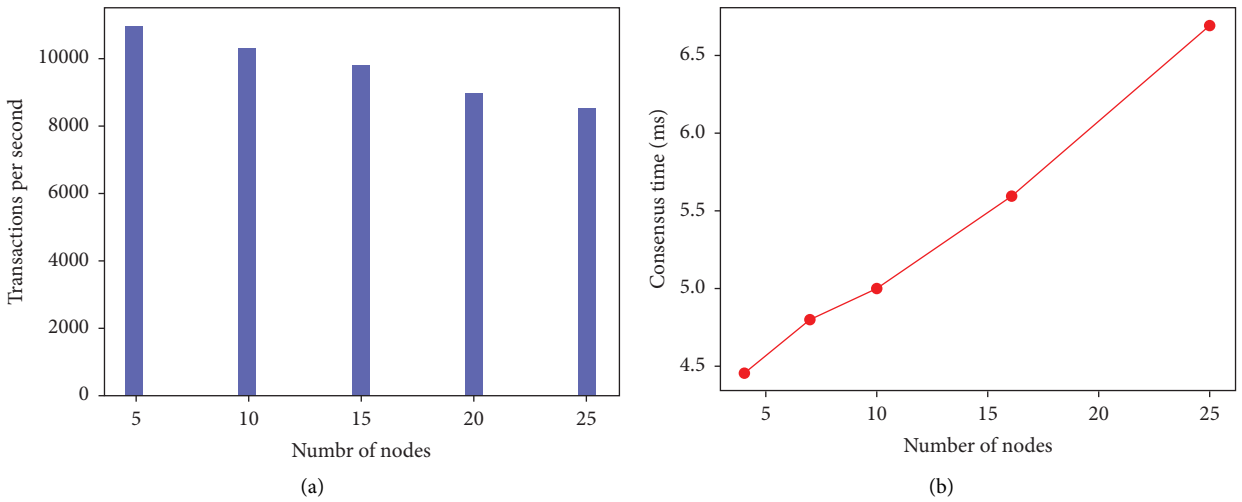

Figure 8: Performance evaluation of blockchain. (a) Throughput. (b) Consensus time.

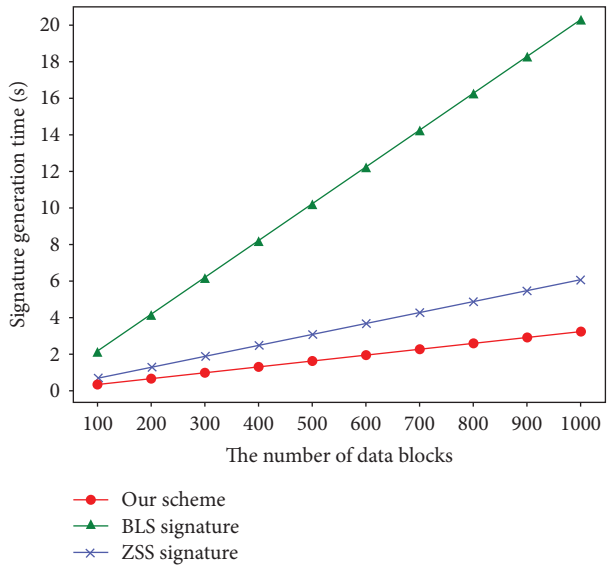

Figure 9: Comparison of signature generation time.

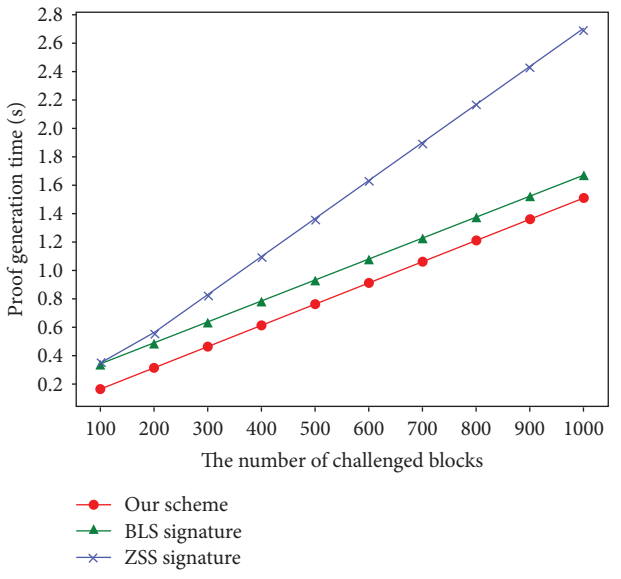

Figure 10: Comparison of proof generation time.

and modulus operations, and the amount of data to be calculated is small, so our scheme has better performance.

Then we simulate the performance of the three schemes in the verification phase. Besides, we also compare the performance of our scheme in the verification phase without the cuckoo filter. The comparison result is shown in Figure 11. The BLS signature has the worst performance due to the need to calculate many exponential operations. Without the cuckoo filter, our scheme's performance is worse than the ZSS signature, since the ZSS signature has only bilinear mapping operations in the verification phase, and only a series of addition and multiplication operations are calculated. However, with the introduction of the cuckoo filter, the complex signature verification process is simplified to a simple cuckoo filter lookup process; the verification process only takes a few

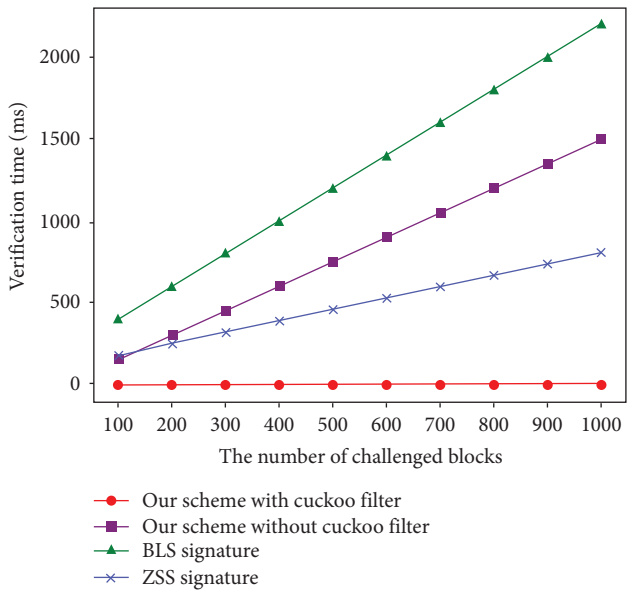

Figure 11: Comparison of verification time.

milliseconds, which is much lower than other methods. When users want to verify the data integrity, they only need to perform a lookup operation of the cuckoo filter to verify whether the signatures returned by the CSP are in the filter. Therefore, compared with the traditional signature verification process, our verification method's time complexity is just $O(1)$, which has a clear advantage in terms of efficiency.

11. Conclusion

With the rapid popularity of cloud storage and the rapid expansion of data on the cloud, ensuring the integrity of the data on the cloud has also become a classic topic. In this paper, we propose an efficient cloud data integrity verification scheme based on blockchain. In our scheme, we use the blockchain network to solve some shortcomings of the traditional centralized audit and improve the efficiency and security of the scheme. On the other hand, based on the assumption of SIS problem, our scheme can resist the threat of quantum computing, and by combining lattice signature and cuckoo filter, we also simplify the user verification process, solving part of the problem of the insufficient computing power of users. The proposed scheme's performance is evaluated, and the result shows that the scheme is provably efficient. In future work, the closer combination of blockchain and integrity verification scheme needs to be explored and more comprehensive characteristics of the scheme need to be satisfied.

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 there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61872134, in part by the Natural Science Foundation of Hunan Province under Grant 2018JJ2062, in part by Science and Technology Development Center of the Ministry of Education under Grant 2019J01020, in part by the 2011 Collaborative Innovative Center for Development and Utilization of Finance and Economics Big Data Property, Universities of Hunan Province, in part by the National Key Research and Development Program of China (2017YFC1703306), and in part by the School Level Project of Hunan University of Chinese Medicine (2018GL01).

References

[1] S. Nepal, S. Chen, J. Yao, and D. Thilakanathan, "DIaaS: Data Integrity as a Service in the Cloud," in *Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing*, pp. 308–315, IEEE, Washington, DC, USA, 2011.

[2] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, pp. 199–212, ACM, New York, NY, USA, 2009.

[3] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pp. 598–609, ACM, New York, NY, USA, 2007.

[4] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: beyond bitcoin," *Applied Innovation*, vol. 71, no. 2, pp. 6–10, 2016.

[5] W. S. H. M. W. Ahmad, N. A. M. Radzi, F. S. Samidi et al., "5G technology: towards dynamic spectrum sharing using cognitive radio networks," *IEEE Access*, vol. 8, pp. 14460–14488, 2020.

[6] J. Su, R. Xu, S. Yu, B. Wang, and J. Wang, "Idle slots skipped mechanism based tag identification algorithm with enhanced collision detection," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 5, pp. 2294–2309, 2020.

[7] J. Su, R. Xu, S. Yu, B. Wang, and J. Wang, "Redundant rule detection for software-defined networking," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 6, pp. 2735–2751, 2020.

[8] X. Jiang, J. Yu, J. Yan, and R. Hao, "Enabling efficient and verifiable multi-keyword ranked search over encrypted cloud data," *Information Sciences*, vol. 403-404, pp. 22–41, 2017.

[9] Z. Zhou, Q. M. J. Wu, Y. Yang, and X. Sun, "Region-level visual consistency verification for large-scale partial-duplicate image search," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 16, no. 2, pp. 1–25, 2020.

[10] Z. Zhou, Y. Mu, and Q. M. J. Wu, "Coverless image steganography using partial-duplicate image retrieval," *Soft Computing*, vol. 23, no. 13, pp. 4927–4938, 2019.

[11] G. Ateniese, R. Di 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*, pp. 1–10, ACM, New York, NY, USA, 2008.

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

[13] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.

[14] E. Zhou, Z. Li, H. Guo, and Y. Jia, "An improved data integrity verification scheme in cloud storage system," *Acta Electronica Sinica*, vol. 42, no. 1, pp. 150–154, 2014.

[15] Y. Zhu, H. Wang, Z. Hu et al., "Dynamic audit services for integrity verification of outsourced storages in clouds," ", ACM, in *Proceedings of the 2011 ACM Symposium on Applied Computing*, pp. 1550–1557, ACM, New York, NY, USA, 2011.

[16] Y. Zhu, G. J. Ahn, H. Hu et al., "Dynamic audit services for outsourced storages in clouds," *IEEE Transactions on Services Computing*, vol. 6, no. 2, pp. 227–238, 2011.

[17] 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, 2012.

[18] J. Ni, Y. Yu, Y. Mu, and Q. Xia, "On the security of an efficient dynamic auditing protocol in cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2760-2761, 2013.

[19] G. Xu, Y. Bai, C. Yan et al., "Check algorithm of data integrity verification results in Big data storage," *Journal of Computer Research and Development*, vol. 54, no. 11, pp. 2487–2496, 2017.

[20] 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, 2019.

[21] A. Li, S. Tan, and Y. Jia, "A method for achieving provable data integrity in cloud computing," *The Journal of Supercomputing*, vol. 75, no. 1, pp. 92–108, 2019.

[22] H. Zhu, Y. Yuan, Y. Chen et al., "A secure and efficient data integrity verification scheme for cloud-IoT based on short signature," *IEEE Access*, vol. 7, pp. 90036–90044, 2019.

[23] K. Huang, M. Xian, S. Fu, and J. Liu, "Securing the cloud storage audit service: defending against frame and collude attacks of third party auditor," *IET Communications*, vol. 8, no. 12, pp. 2106–2113, 2014.

[24] Y. Wu, X. Lin, X. Lu, J. Su, and P. Chen, "A secure light-weight public auditing scheme in cloud computing with potentially malicious third-party auditor," *IEICE Transactions on Information and Systems*, vol. E99.D, no. 10, pp. 2638–2642, 2016.

[25] D. Xiao, L. Yang, B. Sun, and S. Zheng, "Provable data possession system for realistic cloud storage environments," *Journal of Software*, vol. 27, no. 9, pp. 2400–2413, 2016.

[26] X. Boyen, "Lattice mixing and vanishing trapdoors: a framework for fully secure short signatures and more," *Public Key Cryptography-PKC 2010*, vol. 6056, pp. 499–517, 2010.

[27] D. Micciancio and C. Peikert, "Trapdoors for lattices: simpler, tighter, faster, smaller," *Advances in Cryptology-EUROCRYPT 2012*, vol. 7237, pp. 700–718, 2012.

[28] L. Ducas and D. Micciancio, "Improved short lattice signatures in the standard model," *Advances in Cryptology-CRYPTO 2014*, vol. 8616, pp. 335–352, 2014.

[29] J. Alperin-Sheriff, "Short signatures with short public keys from homomorphic trapdoor functions," *IACR International Workshop on Public Key Cryptography*, vol. 2015, pp. 236–255, 9020.

[30] D. Yang, C. Xu, L. Xu, and X. Zhang, "Identity-based signature scheme over ideal lattices," *Journal of Cryptologic Research*, vol. 2, no. 4, pp. 26–36, 2015.

[31] J. Zhang, Y. Chen, and Z. Zhang, "Programmable hash functions from lattices: short signatures and IBEs with small key sizes," *Advances in Cryptology-CRYPTO 2016*, vol. 9816, pp. 303–332, 2016.

[32] H. Tian, F. Zhang, and B. Wei, "A lattice-based partially blind signature," *Security and Communication Networks*, vol. 9, no. 12, pp. 1820–1828, 2016.

[33] Z. Wang, X. Chen, and P. Wang, "Adaptive-ID secure identity-based signature scheme from lattices in the standard model," *IEEE Access*, vol. 5, pp. 20791–20799, 2017.

[34] Y.-L. Gao, X.-B. Chen, Y.-L. Chen, Y. Sun, X.-X. Niu, and Y.-X. Yang, "A secure cryptocurrency scheme based on post-quantum blockchain," *IEEE Access*, vol. 6, pp. 27205–27213, 2018.

[35] T. Hardjono and N. Smith, "Cloud-based commissioning of constrained devices using permissioned blockchains," in *Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security*, pp. 29–36, New York, NY, USA, 2016.

[36] E. Heilman, F. Baldimtsi, and S. Goldberg, "Blindly signed contracts: anonymous on-blockchain and off-blockchain bitcoin transactions," *Financial Cryptography and Data Security*, vol. 9604, pp. 43–60, 2016.

[37] X. Liang, S. S. Shetty, D. Tosh et al., *ProvChain: Blockchain-based Cloud Data Provenance*, Blockchain for Distributed Systems Security, New York, NY, USA, 2019.

[38] L. Zhu, Y. Wu, K. Gai, and K.-K. R. Choo, "Controllable and trustworthy blockchain-based cloud data management," *Future Generation Computer Systems*, vol. 91, pp. 527–535, 2019.

[39] D. Yue, R. Li, Y. Zhang, W. Tian, and C. Peng, "Blockchain based data integrity verification in P2P cloud storage," in *Proceedings of the 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 561–568, IEEE, New York, NY, USA, 2018.

[40] H. Wang, Q. Wang, and D. He, *Blockchain-Based Private Provable Data Possession*, IEEE Transactions on Dependable and Secure Computing, New York, NY, USA, 2019.

[41] Y. Zhang, C. Xu, X. Lin, and X. S. Shen, "Blockchain-based public integrity verification for cloud storage against procrastinating auditors," *IEEE Transactions on Cloud Computing*, vol. 23, p. 1, 2019.

[42] J. Tian and T. Li, "Data integrity verification based on model cloud federation of TPA," *Journal on Communications*, vol. 39, no. 8, pp. 113–124, 2018.

[43] B. Liu, X. L. Yu, S. Chen, X. Xu, and L. Zhu, "Blockchain based data integrity service framework for IoT data," in *Proceedings of the 2017 IEEE International Conference on Web Services (ICWS)*, pp. 468–475, New York, NY, USA, 2017.

[44] P. Wei, D. Wang, Y. Zhao, S. K. S. Tyagi, and N. Kumar, "Blockchain data-based cloud data integrity protection mechanism," *Future Generation Computer Systems*, vol. 102, pp. 902–911, 2020.

[45] S. Nakamoto, "Bitcoin: A Peer-To-Peer Electronic Cash System," *Manubot*, vol. 23, 2019.

[46] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher, "Cuckoo filter: practically better than bloom," in *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies*, pp. 75–88, ACM, New York, NY, USA, 2014.

[47] L. Chen, S. Jordan, Y. K. Liu et al., *Report on Post-quantum Cryptography*, National Institute of Standards and Technology, New York, NY, USA, 2016.