Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 8772700, 14 pages
https://doi.org/10.1155/2023/8772700

Research Article

Post-Quantum Privacy-Preserving Provable Data Possession Scheme Based on Smart Contracts

Zhengwen Li,[1] **Tengjiao Zhang**,[2] **De Zhao**,[2] **Yaxing Zha**,[3] **and Jun Sun**[4]

[1]*School of Cyberspace Security, Beijing University of Posts and Telecommunications, 100876, China*
[2]*School of Information Engineering, Beijing Institute of Graphic Communication, 102600, China*
[3]*Research and Development Center of Transport Industry of Network Security Technologies, China Communication Information Technology Group Co., Ltd, 100088, China*
[4]*China Industrial Control Systems Cyber Emergency Response Team, 100040, China*

Correspondence should be addressed to Jun Sun; jasonsunjun@163.com

Received 18 June 2022; Revised 18 July 2022; Accepted 6 October 2022; Published 2 February 2023

Academic Editor: Amrit Mukherjee

Provable data possession (PDP) is a crucial means of protecting the integrity of data in the domain of cloud storage. In the post-quantum era, the PDP scheme that uses lattices relies too heavily on the third-party auditor (TPA), which is not entirely trustworthy and is easily affected by a single point of failure. Moreover, the scheme often leads to the leakage of private user data while attempting to satisfy the demand for public verification. In response to the above problems, this paper designs a protocol for post-quantum privacy-preserving PDP and uses it to develop a scheme based on smart contracts. The proposed scheme has the characteristics of being post quantum and can satisfy the demand for public verification while preserving user privacy. The property of noninteraction of the protocol can reduce transaction fees incurred owing to the frequent operation of the blockchain, and the smart contract with a deposit mechanism can ensure fair payments to all parties. The results of a theoretical analysis and experiments show that the proposed scheme is highly secure and efficient.

1. Introduction

Cloud storage has emerged as the crucial infrastructure for data storage with the rapid development of cloud computing, big data, the Internet of Things, the mobile Internet, and artificial intelligence in recent years. It can deliver services on demand with just-in-time capacity and costs and eliminates the need for users to buy and manage their own infrastructure for data storage. This provides users with agile, global, and durable—"anytime, anywhere"—access to data [1].

While cloud storage is widely used, it poses new security threats to cloud data because users lose physical control over their data [2]. An untrusted cloud storage provider (CSP) may reclaim storage space for economic reasons by deleting infrequently accessed data or even concealing events of data loss to avoid damage to its reputation. In addition, server failures, power outages, and security attacks occasionally occur to seriously compromise the integrity and availability of data in the cloud. Owing to various security-related incidents, such as the service downtime of cloud storage and the leakage of private data [3, 4], the integrity of data in cloud storage is a cause for concern for users and has greatly restricted the development of this technology.

Researchers have proposed efficient schemes of verification to protect the integrity of data in cloud storage. Data integrity can be verified by computing a small number of data blocks without downloading all the data in the cloud to the local server. Provable data possession (PDP) is the most common scheme to verify the integrity of data. The user in this scheme divides the uploaded data into blocks of equal size and then calculates the tag of each block. The data blocks and the tags are then sent to the CSP. When data integrity needs to be verified, the user randomly selects a few data

blocks, asks the CSP to calculate a short proof based on their tags, and then verifies the proof returned by it to determine whether it stores all of the user's data. Ateniese et al. [5] proposed a PDP scheme that uses RSA-based homomorphic verifiable tags to aggregate signatures, thus reducing the burden of computation and I/O access to significantly improve the efficiency of verification. Wang et al. [6] used the BLS [7] to design a scheme to verify data integrity. They constructed a Merkle hash tree (MHT) that supports dynamic data operations, where the security of the system depends on the computational Diffie–Hellman (CDH) problem. Curtmola et al. [8] extended PDP to the case of replicas and proposed multiple-replica provable data possession (MR-PDP) that can reduce the overhead of all replicas due to verification to roughly the same as that of a single replica.

It is essential for users with limited computing resources and capabilities to provide public verification. They can designate external auditors to check the integrity of data in cloud storage when needed. A variant of PDP has been proposed [5] to support public verification. It allows anyone (not just the data owner) to challenge and check the data ownership of the CSP. Wang et al. [9] introduced a third-party auditor (TPA), instead of users, that regularly verifies the integrity of remote data without storing all data. Once verification fails, it can immediately notify the user that the data are damaged and take timely measures to avoid more significant losses. On the contrary, the results of an audit submitted by the TPA ensure some objectivity and fairness, where this is conducive to the accountability of users after disputes with the CSP. Therefore, most subsequent PDP schemes [10–15] tend to include a TPA.

However, we cannot base data security on the assumption of a fully trustworthy TPA. A malicious TPA may corrupt and leak user data or deceive users with fake audits. Some researchers have used the blockchain instead of the TPA in the PDP scheme. Because the blockchain is open, transparent, and tamper resistant, it provides users with a completely trusted third-party audit, and users can supervise the entire process. Zhang et al. [16] proposed a blockchain-based fair payment framework for outsourcing services in cloud computing and used it to build a blockchain-based PDP scheme. Chen et al. [17] proposed a decentralized blockchain-based PDP scheme that uses a rank-based MHT to support the dynamic operation of outsourced data, uses the blockchain to record all transaction behaviors, and deploys smart contracts combined with the deposit mechanism to ensure the automatic execution of the protocol and fair payments. Chen et al. [18] proposed the first decentralized system for proofs of data retrievability and replication—BOSSA, which is incentive-compatible for each party and realizes automated auditing by using smart contracts on the Ethereum blockchain. Wang et al. [19] proposed the concept of noninteractive public provable data possession and used it to design a blockchain-based smart contract for the public auditing of cloud storage. This scheme is based on bilinear pairing, and its security relies on the CDH problem and the discrete logarithm problem.

Since public verification allows people other than users to verify data integrity, it is necessary to ensure that the audit process preserves user privacy such that the auditors cannot obtain any original user data based on the information collected. Encrypting data before uploading them is one way to protect user privacy but can be used only as a supplement and does not by itself prevent data from "flowing" out if appropriate protocols are not designed. Wang et al. [9] used random masking technology to solve the problem of privacy leakage. Two random parameters are introduced to the process of proof generation to prevent the adversary from obtaining private data by solving linear equations based on the content of the message. References [20–22] used the same idea to protect user privacy. Tong et al. [23] used private information retrieval and the tag repacking technique to enable the TPA to check the integrity of data without violating the privacy of the user data and the query pattern.

With the development of quantum computing [24] in recent years, the hard problems of traditional cryptography, such as integer factorization and the discrete logarithm, can be solved by quantum computers in polynomial time. The security of all information systems designed based on these problems is thus seriously affected, and a PDP scheme is needed to resist attacks based on quantum computing. Lattice cryptography has the advantages of a strong proof of security with worst-case hardness, simple construction, and easy implementation [25]. It has a high speed of calculation and low communication overhead and can be used to construct various cryptographic algorithms and applications. It is thus considered the most efficient and promising method of post-quantum cryptography. Zhang and Xu [26] proposed a post-quantum secure cloud storage system that supports privacy preservation and public auditing. Its security is based on the hardness of inhomogeneous small integer solution (ISIS) on lattices. The lattice signatures generated by a function with preimage sampling are used for random masking to ensure that the TPA cannot acquire knowledge about the data. Yang et al. [27] proposed an identity-based model to audit the integrity of data that does not rely on a public key infrastructure, and its security is based on ring learning with errors (RLWE). Tan et al. [28] proposed a provable data integrity scheme based on lattices in cloud storage that uses the homomorphic signature technique on lattices [29] and supports dynamic data operation and batch verification. Based on the work in [28], Tan et al. [30] used random masking technology to achieve privacy preservation such that the TPA cannot obtain the original data.

In the context of the post-quantum era, the PDP scheme based on lattice cryptography relies too heavily on the TPA, which is not entirely trustworthy and is easily affected by a single point of failure. Moreover, this scheme often leads to the leakage of private user data in the quest to satisfy the demand for public verification. In response to these problems, this paper designs a post-quantum privacy-preserving PDP protocol and uses it to propose a privacy-preserving scheme based on smart contracts. The proposed scheme is post quantum and can be used for public verification while ensuring the preservation of private user data. The characteristic of noninteraction of the protocol can reduce transaction fees incurred by the frequent

operation of the blockchain, and the smart contract with a deposit mechanism can ensure fair payments to all parties.

The main contributions of this research can be summarized as follows:

(1) We design a post-quantum privacy-preserving PDP protocol. It uses linearly homomorphic signatures on lattices to calculate the signatures of different data blocks and generate a short proof. Data possession can be verified through this proof. The security of the scheme depends on the hardness of the inhomogeneous small integer solution on the lattices, thereby ensuring resistance against quantum computing-based attacks. The protocol also uses random masking technology to add random variables to the generated proof so that the verifier cannot obtain the user's data by solving a linear system of equations composed of multiple pieces of proof, thereby preserving user privacy

(2) We propose a post-quantum privacy-preserving PDP scheme based on smart contracts. The audit process is open and transparent and is regularly and automatically verified by deploying smart contracts on the blockchain. The smart contract stipulates the rights and obligations of all parties involved. Once the conditions have been triggered, the contract is automatically executed to ensure fair payments to all parties, and this reduces the cost of handling disputes. In addition, the cost of transactions decreases significantly because the participants do not need to interact in the "challenge–response" phase owing to frequent interactions on the blockchain, and this increases the verifier's enthusiasm for executing smart contracts

(3) We analyze the correctness and capability of privacy preservation of the proposed scheme and prove that it is secure under the random oracle model through interactive analysis of the construction of a series of games. Finally, the practicability and efficiency of the proposed scheme are verified through comparative experiments with prevalent methods in the area

The rest of the paper is organized as follows. Section 2 recalls some preliminaries used in our scheme. Section 3 defines the model of our scheme, gives the formal definition, and presents the concrete construction. Section 4 provides the security proof of the protocol. Section 5 evaluates the performance of our scheme. Finally, we give a conclusion in Section 6.

2. Preliminary

2.1. Provable Data Possession. To reduce storage overhead, users store their original data in the cloud instead of doing so locally. Therefore, the CSP is needed to prove the possession of data to periodically ensure their integrity. The TPA is introduced to the PDP scheme to verify the integrity of the data on behalf of the users to reduce the overheads incurred by them due to computation and communication. During the verification process, the CSP needs to respond to the challenge initiated by the TPA and generate a proof for it. The result of verification is returned to the user.

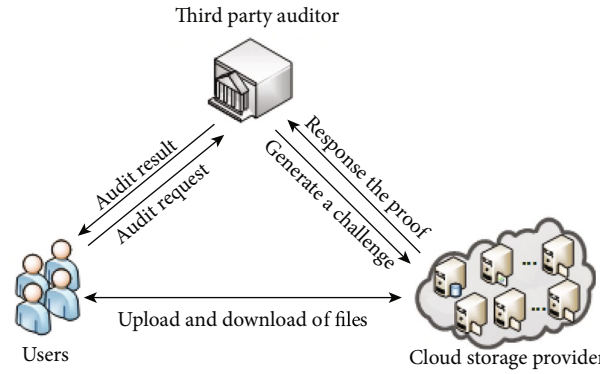

Figure 1: General system model of PDP.

According to the basic definition of PDP, a general system model of it is shown in Figure 1. The participating entities include the users, CSP, and TPA.

The PDP scheme consists of five algorithms: key generation, tag generation, challenge generation, proof generation, and proof verification. The description of each follows.

(1) $\text{KeyGen}(1^{\lambda}) \longrightarrow (\text{pk}, \text{sk})$: the key generation algorithm is run by the user. The input to it is the system security parameter λ and the output is a public–private key pair (pk, sk)

(2) $\text{TagGen}(\text{pk.sk}, m_i) \longrightarrow T_i$: the tag generation algorithm is run by the user. The inputs are the public–private key pair (pk, sk) and the data block m_i, and the output is the signature T_i of m_i. The user sends (m_i, T_i) to the CSP

(3) $\text{GenChallenge}(\text{Id}_m, \tau, Q) \longrightarrow \text{chal}$: the challenge generation algorithm is run by the TPA. The inputs to it are the identification information Id_m, a random number $\tau \in \mathbb{Z}_q^*$, and a challenge set Q, and the output is the challenge chal

(4) $\text{GenProof}(m_i, T_i, \text{chal}) \longrightarrow \rho$: the proof generation algorithm is run by the CSP. The input to it consists of the data block m_i, the signature T_i, and the challenge chal, and the output is the proof ρ. The CSP obtains the proofs μ and σ by using homomorphic aggregation technology to process m_i and T_i, respectively, and the CSP returns the result of aggregation $\rho = \{\mu, \sigma\}$ to the TPA

(5) $\text{ProofCheck}(\rho) \longrightarrow \{\text{"true"}, \text{"false"}\}$: the proof verification algorithm is run by the TPA. The input to it is the proof $\rho = \{\mu, \sigma\}$. If the proof is verified, the output is true; otherwise, the output is false

The above algorithms constitute the general PDP scheme. To satisfy the requirements of different application

scenarios, researchers have improved and supplemented this scheme.

2.2. Lattice-Based Homomorphic Signature

Notation 1. For any integer $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integer modulo q. If q is a prime number, $\mathbb{Z}_q$ is a field and is denoted by $\mathbb{F}_q$. We let $\mathbb{Z}_q^{m\times n}$ denote the set of $m \times n$ matrices with entries $\mathbb{Z}_q$. We use standard big-O notation to classify the growth of functions and say that if $f(n) = O(g(n) \cdot \log^c n)$, we let $\text{poly}(n)$ denote an unspecified function $f(n) = O(n^c)$ for some constant c. A negligible function, denoted generically by $\text{negl}(n)$, is an $f(n)$ such that $f(n) = o(n^{-c})$ for every fixed constant c. We say that a probability is overwhelming if it is $1 - \text{negl}(n)$. The base 2 logarithm is denoted $\log x$.

Definition 2. Lattice: an n-dimensional lattice of rank $k \leq n$ is

$$\Lambda = \mathcal{L}(B) = \left\{ Bc : c \in \mathrm{Z}^k \right\}, \quad B \in \mathbb{R}^{n\times k}, \tag{1}$$

where the k columns $b_1, \cdots, b_k \in \mathbb{R}^n$ of the basis B are linearly $f(n) = \tilde{O}(g(n))$ independent. $\|B\|$ denotes the length of the longest vector in B, i.e., $\max_{1\leq i\leq k} \|b_i\|$, and $\tilde{B} = \{\tilde{b}_1, \cdots, \tilde{b}_k\}$ denotes the Gram–Schmidt orthogonalization of the vectors $b_1, \cdots, b_k$.

For any integer $q \geq 2$ and any $A \in \mathbb{Z}_q^{m\times n}$, we define

$$\begin{aligned} \Lambda_q^{\perp}(A) &\coloneqq \{e \in \mathbb{Z}^n : A \cdot e = 0 \bmod q\}, \\ \Lambda_q^{u}(A) &\coloneqq \{e \in \mathbb{Z}^n : A \cdot e = u \bmod q\}. \end{aligned} \tag{2}$$

The lattice $\Lambda_q^u(A)$ is a coset of $\Lambda_q^{\perp}(A)$, namely, $\Lambda_q^u(A) = \Lambda_q^{\perp}(A) + t$ for any t such that $A \cdot t = u \bmod q$.

Lemma 3 ([31] Lemma 7.1). *Let Λ be an m-dimensional lattice. There is a deterministic polynomial-time algorithm that, given an arbitrary basis of Λ and a full-rank set $S = \{s_1, \cdots, s_m\}$ in Λ, returns a basis T of Λ satisfying*

$$\begin{aligned} \left\| \tilde{T} \right\| &\leq \left\| \tilde{S} \right\|, \\ \|T\| &\leq \|S\| \cdot \frac{\sqrt{m}}{2}. \end{aligned} \tag{3}$$

Theorem 4 ([32] Theorem 3.2). *Let q, m, n be positive integers with $q \geq 2$ and $m \geq 6n \log q$. There is a probabilistic polynomial-time algorithm $\text{TrapGen}(q, n, m)$ that outputs $(A \in \mathbb{Z}_q^{n\times m}, S \in \mathbb{Z}^{m\times m})$ such that A is statistically close to a uniform in $\mathbb{Z}_q^{n\times m}$ and S is a basis for $\Lambda_q^{\perp}(A)$ that satisfies*

$$\begin{aligned} \left\| \tilde{S} \right\| &\leq O\left(\sqrt{n \cdot \log q} \right), \\ \|S\| &\leq O(n \cdot \log q), \end{aligned} \tag{4}$$

with all but negligible probability in n.

Definition 5. Gaussian distributions: let L be a subset of $\mathbb{Z}^n$. For any vector $c \in \mathbb{R}^n$ and any positive $\sigma \in \mathbb{R}^+$, let $\rho_{\sigma,c}(x) \coloneqq \exp\left((-\pi\|x - c\|^2)/\sigma^2\right)$ be a Gaussian function on $\mathbb{R}^n$ with center c and parameter σ. Let $\rho_{\sigma,c}(L) \coloneqq \sum_{x\in L} \rho_{\sigma,c}(x)$ be the discrete integral of $\rho_{\sigma,c}$ over L, and let $D_{L,\sigma,c}$ be the discrete Gaussian distribution over L with center c and parameter σ. For all $y \in L$, $D_{L,\sigma,c}(y) = \rho_{\sigma,c}(y)/\rho_{\sigma,c}(L)$. For notational convenience, $\rho_{\sigma,0}$ and $D_{L,\sigma,0}$ are abbreviated as ρ_σ and $D_{L,\sigma}$, respectively.

Gentry et al. [33] constructed algorithms to sample from discrete Gaussian distributions.

Theorem 6.

(a) *There is a probabilistic polynomial-time algorithm SampleGaussian that, given a basis T of an n-dimensional lattice Λ, a parameter $\sigma \geq \|\tilde{T}\| \cdot \omega(\sqrt{\log n})$, and a center $c \in \mathbb{R}^n$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda,\sigma,c}$*

(b) *There is a probabilistic polynomial-time algorithm SamplePre that, given a basis T of an n-dimensional lattice Λ, a parameter $\sigma \geq \|\tilde{T}\| \cdot \omega(\sqrt{\log n})$, and a vector $t \in \mathbb{R}^n$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda+t,\sigma}$*

Definition 7 ([34] Definition 3.1). Smoothing parameter: for an n-dimensional lattice Λ and positive real $\varepsilon > 0$, the smoothing parameter $\eta_\varepsilon(\Lambda)$ of Λ is defined to be the smallest positive s such that $\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \varepsilon$, where Λ^* is the dual lattice of Λ.

The critical property of the smoothing parameter is that if $\sigma > \eta_\varepsilon(\Lambda)$, then every coset of Λ has roughly equal mass under the distribution $D_{\Lambda,\sigma,c}$.

Lemma 8 ([34] Lemma 4.4). *Let Λ be an n-dimensional lattice. Let T be a basis for Λ, and suppose $\sigma \geq \|\tilde{T}\| \cdot \omega(\sqrt{\log n})$. Then, for any $c \in \mathbb{R}^n$, we have*

$$Pr\left\{ \|x - c\| > \sigma\sqrt{n} : x \xleftarrow{R} D_{\Lambda,\sigma,c} \right\} \leq negl(n). \tag{5}$$

Let Λ_1 and Λ_2 be two lattices in $\mathbb{Z}^n$ such that $\Lambda_1 + \Lambda_2 = \mathbb{Z}^n$. We show that a discrete Gaussian sampling from Λ_1 is close to uniform in $\mathbb{Z}^n/\Lambda_2$ by the following lemma.

Lemma 9 ([29] Lemma 3.9). *Let Λ_1 and Λ_2 be n-dimensional lattice such that $\Lambda_1 + \Lambda_2 = \mathbb{Z}^n$. For any $\varepsilon \in (0, 1/2)$ and $\sigma \geq \eta_\varepsilon(\Lambda_1 \cap \Lambda_2)$ and any $c \in \mathbb{R}^n$, the distribution $\mathcal{D}_{\Lambda_1,\sigma,c} \bmod \Lambda_2$ is within statistical distance at most 2ε of the uniform distribution over $(\Lambda_1 + \Lambda_2)/\Lambda_2$.*

Definition 10. The inhomogeneous small integer solution (ISIS) problem is as follows: given $q \in \mathbb{Z}$, $\beta \in \mathbb{R}$, $A \in \mathbb{Z}_q^{n\times m}$, u

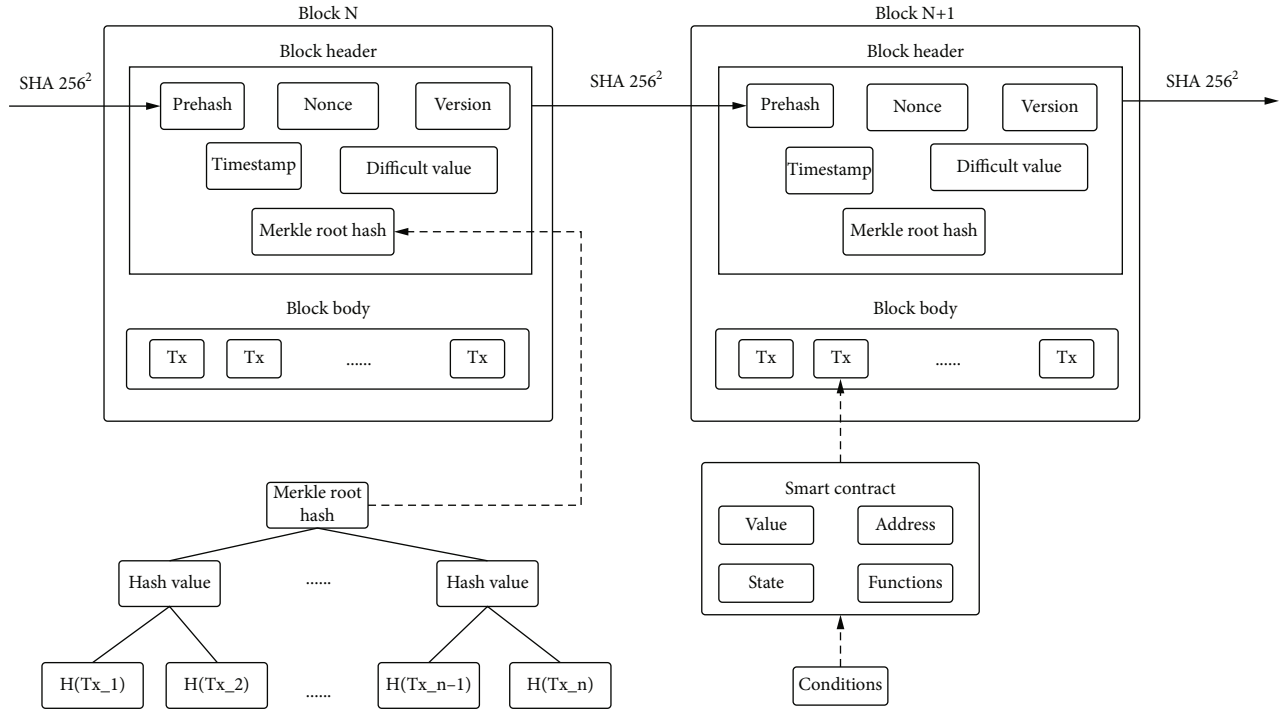

Figure 2: Structure of the blockchain.

$\in \mathbb{Z}_q^n$, find a nonzero vector $v \in \mathbb{Z}^m$ such that $Av = u(\mathrm{mod}\ q)$ and $\|v\| \leq \beta$.

Lemma 11 ([33] Proposition 5.7). *For any poly-bounded $m, \beta = poly(n)$ and any prime $q \geq \beta \cdot \omega(\sqrt{n \cdot \log n})$, the average-case problem ISIS is as hard as that of approximating the SIVP problem in the worst case.*

2.3. Blockchain and Smart Contracts. The blockchain is managed by a peer-to-peer network for maintaining a secure and decentralized distributed ledger of transactions, where nodes collectively adhere to a protocol to communicate and validate new blocks. It is best known for its vital role in cryptocurrency systems such as Bitcoin [35]. The innovation of the blockchain guarantees the fidelity and security of data records and generates trust without the need for a trusted third party.

The difference between the blockchain and other databases lies mainly in how the data are composed. The data are stored in blocks, which are generated in chronological order and connected to a chain. This data structure forms an immutable data timeline, with each block in the chain being added with a precise timestamp. As shown in Figure 2, each block contains a block header and a block body. The block header contains the hash value of the previous block, a version number, a random value, a timestamp, a Merkle root hash, and a difficulty value. The block body contains all transactions generated during block creation. Each block in the blockchain is identified by a hash value obtained by twice executing the SHA256 algorithm of the block header. Each block can find its previous block by using the hash value contained in its block header. Any changes to the data in the block lead to a series of changes in subsequent blocks, and distributed nodes run a consensus protocol to synchronously update the hash chain. Therefore, the blockchain has the characteristics of decentralization, transparency, openness, immutability, and traceability.

The smart contract [36] is a piece of code in the blockchain in which the logic of the code defines the contents of the contract. Smart contracts operate under a set of conditions agreed upon by all parties involved; when these conditions are triggered, the contents of the contract are automatically executed. Say that a tenant wants to use a smart contract to lease an apartment from a landlord. First, the tenant and the landlord negotiate details of the contract, such as the duration of the lease, rent, deposit, and the terms of compensation. They then write the content as agreed by both parties into the smart contract and deploy it on the blockchain. The contract takes effect within the specified time. It is automatically executed according to its contents such that this reduces the costs of notarization, mediation, and litigation in case disputes arise.

We can say that the blockchain provides a trusted execution environment for smart contracts, which in turn extends the application of the blockchain. Smart contracts have been used in many fields, such as electronic voting [37] and insurance [38], and have excellent prospects for further use.

3. Our Scheme

3.1. System Model. The system model of a post-quantum privacy-preserving provable data possession scheme based on smart contracts contains three entities: the data owner, the cloud storage provider, and the verifier (see Figure 3).

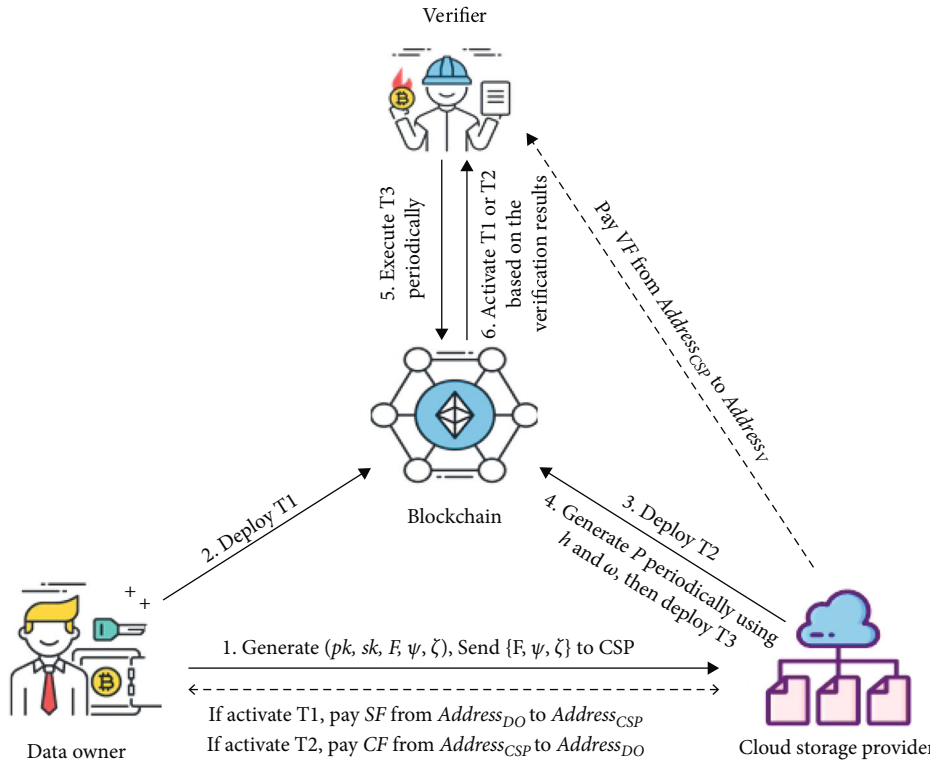

Figure 3: System model.

(1) Data owner (DO): to save the cost of local storage and use data conveniently and flexibly, cloud storage users pay a certain fee and store their data on a remote server owned by the cloud storage provider

(2) Cloud storage provider (CSP): it provides computing, storage, and network bandwidth for the DO. The CSP needs to periodically perform PDP to prove that it maintains the integrity of data. If verification fails, it pays the DO a certain fee by way of compensation

(3) Verifier: this is the executor of the proof verification algorithm. Because the scheme is publicly verifiable, all members of the blockchain can theoretically act as verifiers, usually as third-party miners. Verifiers get a certain compensation by executing smart contracts deployed on the blockchain

3.2. Formal Definition. Our scheme contains four algorithms: KeyGen, TagGen, ProofGen, and ProofVerify. Each is formally defined below.

(1) $\text{KeyGen}(1^n, k) \longrightarrow (\text{pk}, \text{sk})$: the key generation algorithm is run by the DO. The inputs to it are the security parameter n and the number of data blocks k, and the outputs is a public–private key pair (pk, sk)

(2) $\text{TagGen}(\text{pk}, \text{sk}, F) \longrightarrow (\psi, \zeta)$: the tag generation algorithm is run by the DO. The input to it consists of the public key pk, private key sk, and file F. The outputs are a set of block signatures ψ and a set of block indices ζ

(3) $\text{ProofGen}(F, \psi, \zeta, h, \omega) \longrightarrow \mathscr{P}$: the proof generation algorithm is run by the CSP. The inputs to it are F, a set of block signatures ψ, a set of block indices ζ, the hash value h of the previous block, and timestamp ω. The output is the proof $\mathscr{P}$

(4) $\text{ProofVerify}(\mathscr{P}, \text{pk}) \longrightarrow (\text{SUCCESS}, \text{FALSE})$: the proof verification algorithm is run by the verifier. The inputs are the proof $\mathscr{P}$ and the public key pk. If the proof is verified, the output is SUCCESS; otherwise, the output is FALSE

3.3. Scheme Implementation. We first design a post-quantum privacy-preserving PDP protocol and then combine it with smart contract technology to develop our scheme.

3.3.1. Post-Quantum Privacy-Preserving PDP Protocol. We set a security parameter n and a maximum file block size k. Two hash functions $H_1 : \{0,1\}^* \longrightarrow \{0,1\}^m$ and $H_2 : \{0,1\}^* \longrightarrow F_q^n$ (modeled as a random oracle) are used as well.

KeyGen

(1) Choose two primes $p = 2$ and $q = \text{poly}(n)$ with $q \geq (nkp)^2$. Define $m = \lfloor 6n \log q \rfloor$ and $v = p \cdot \sqrt{m \log q} \cdot \log m$

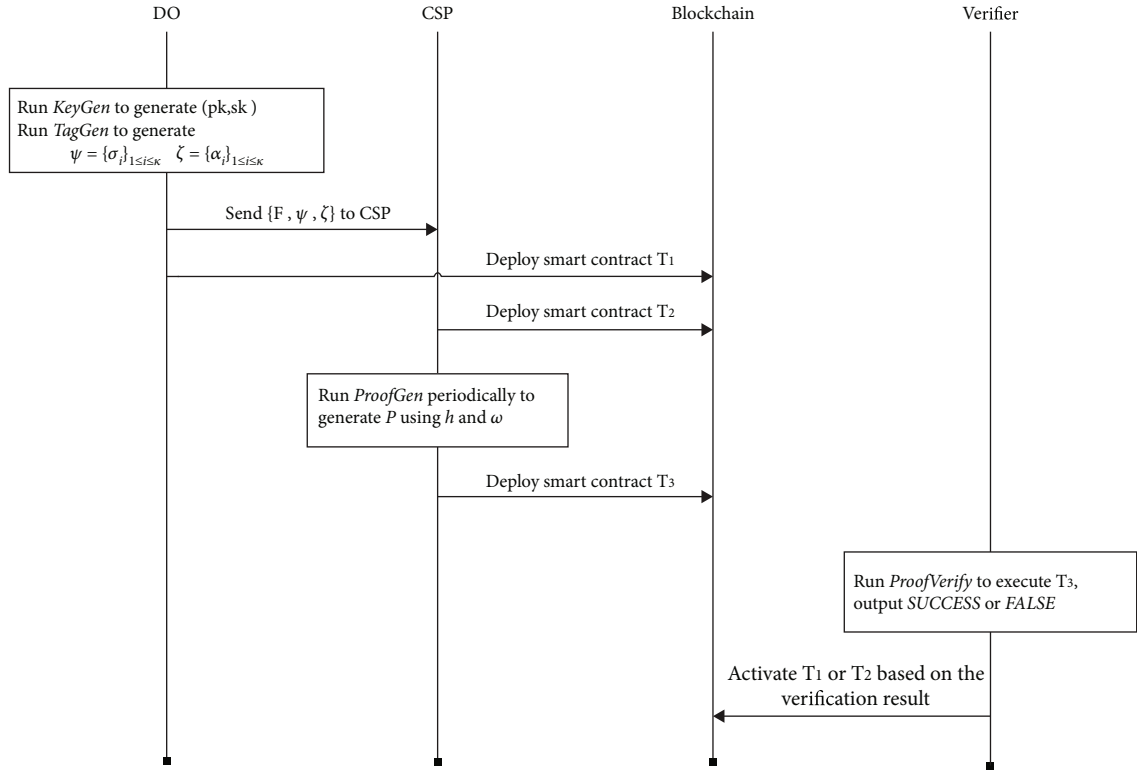

Figure 4: The flow logic of the scheme.

(2) Set $\Lambda_1 = p \cdot \mathbb{Z}^m$. Use TrapGen$(q, n, m)$ to generate a matrix $A \in F_q^{n\times m}$, which results in a lattice $\Lambda_q^{\perp}(A)$ and a short basis T_q of $\Lambda_q^{\perp}(A)$. Define $\Lambda_2 = \Lambda_q^{\perp}(A)$ and $T = p \cdot T_q$, and note that T is a basis of the lattice $\Lambda_1 \cap \Lambda_2 = p \cdot \Lambda_2$

(3) Output the public key $\mathrm{pk} = (\Lambda_1, \Lambda_2, A, v, p, q)$ and the private key $\mathrm{sk} = T$

TagGen

(1) Given the file $F = \{m_i\}_{1\le i\le k}, m_i \in F_p^m$, its fingerprint value can be denoted by the hash $\tau \longleftarrow H_1(F)$

(2) Compute the index $\alpha_i = H_2(\tau \| i) \in F_q^n$ for each block m_i

(3) Compute the vector $t_i \in \mathbb{Z}^m$ such that $t_i \bmod p = m_i$ and $A \cdot t_i \bmod q = \alpha_i$

(4) Compute the signature of m_i:

$$\sigma_i \longleftarrow \mathrm{SamplePre}(\Lambda_1 \cap \Lambda_2, T, t_i, v) \in p\Lambda_2 + t_i \quad (6)$$

(5) Output a set of signatures $\psi = \{\sigma_i\}_{1\le i\le k}$ and a set of block indices $\zeta = \{\alpha_i\}_{1\le i\le k}$

ProofGen

(1) The CSP obtains the header of the newly generated block to obtain the hash value h of the previous block and the corresponding timestamp ω. Because the blockchain uses SHA-256, $h = h_1 \| h_2 \| \cdots \| h_{256}, h_i \in \{0, 1\}$

(2) Define the challenge set as $Q = (c_1, c_2, \cdots, c_k) \in \mathbb{Z}^k$, where $c_i = h_j$ and $i = j \bmod 256$. $c_i \in \{0, 1\}$ satisfies $c_i \in (-p/2, p/2]$

(3) Select two random vectors $R = (r_1, \cdots, r_k) \in \mathbb{Z}_p^k$ and $s \in \mathbb{Z}_p^m$. Compute $s + \sum_{i=1}^k r_i c_i m_i = \mu \in \mathbb{Z}_p^m (\bmod p)$, $\sigma = s + \sum_{i=1}^k r_i c_i \sigma_i$, and $As + \sum_{i=1}^k r_i c_i \alpha_i = \phi \in F_q^n$

(4) Output the proof $\mathcal{P} = \{\mu, \sigma, \phi, h, \omega\}$

ProofVerify

(1) Verify the authenticity of the timestamp ω and the hash value of the previous block h from $\mathcal{P}$. Return FALSE if it fails, and continue if it succeeds

(2) Verify the conditions below. If all conditions are met, output SUCCESS; otherwise, output FALSE:

(a) $\|\sigma\| \le (vk + 1)\sqrt{m}$

(b) $\sigma \bmod p = \mu$

(c) $A \cdot \sigma \bmod q = \phi$

3.3.2. Post-Quantum Privacy-Preserving PDP Scheme Based on Smart Contracts. The DO needs to pay the CSP to purchase storage space in the traditional cloud storage system. If the DO's data are unavailable or have been tampered with,

Table 1: Smart contracts T_1, T_2, and T_3.

Storage contract T_1		Compensation contract T_2		Verification contract T_3	
File's name	FN	File's name	FN	File's name	FN
File's hash	FH	File's hash	FH	File's hash	FH
Upload time	UT	Receive time	RT	Proof	$\mathscr{P}$
Storage fee	SF	Compensation fee	CF	Storage contract	T_1
DO's address	$\text{Address}_{\text{DO}}$	DO's address	$\text{Address}_{\text{DO}}$	Compensation contract	T_2
CSP's address	$\text{Address}_{\text{CSP}}$	CSP's address	$\text{Address}_{\text{CSP}}$	Verification fee	VF
DO's signature	Sign_{DO}	CSP's signature	Sign_{CSP}	CSP's address	$\text{Address}_{\text{CSP}}$
				Verifier's address	$\text{Address}_{\text{V}}$
				CSP's signature	Sign_{CSP}
Contract content: Promise If $\text{ProofVerify}(\text{pk}, \mathscr{P}) \longrightarrow \text{SUCCESS}$ Pay SF from $\text{Address}_{\text{DO}}$ to $\text{Address}_{\text{CSP}}$		Contract content: Promise If $\text{ProofVerify}(\text{pk}, \mathscr{P}) \longrightarrow \text{FALSE}$ Pay CF from $\text{Address}_{\text{CSP}}$ to $\text{Address}_{\text{DO}}$		Contract content: (1) Execute the algorithm ProofVerify If $\text{ProofVerify}(\text{pk}, \mathscr{P}) \longrightarrow \text{SUCCESS}$, then activate the contract T_1 If $\text{ProofVerify}(\text{pk}, \mathscr{P}) \longrightarrow \text{FALSE}$, then activate the contract T_2 (2) Pay VF from $\text{Address}_{\text{CSP}}$ to $\text{Address}_{\text{V}}$	

it is difficult to protect the data rights and obtain economic compensation. On the contrary, the laws and regulations on data security in various countries are not necessarily complete, especially in case of transnational disputes. Legal action also often incurs additional costs in terms of money and time.

The emergence of smart contracts has helped solve the above problems. As smart contracts are immutable and automatically triggered, the contract is executed immediately once all parties have agreed to its contents if all its conditions are met. No one can then change the contents of the contract.

We propose a post-quantum privacy-preserving provable data possession scheme based on smart contracts for cloud storage systems. We provide parties with tamper-resistant integrity verification through smart contracts deployed on the blockchain that pass the deposit mechanism to guarantee fair payment. To protect against dishonest verifiers and ensure the correctness of verifiers when executing smart contracts, a consensus mechanism is needed. This is not discussed in detail in this article.

The flow logic of the scheme is shown in Figure 4 and is described below. The smart contracts used are shown in Table 1.

(1) The DO, CSP, and verifier are registered on the blockchain to obtain the public–private key pair and account addresses $\text{Address}_{\text{DO}}$, $\text{Address}_{\text{CSP}}$, and $\text{Address}_{\text{V}}$. The public–private key pair is used for signing and verifying on the blockchain. The account addresses are generated from the public key to indicate the identity of the transactions. When the scheme is applied to multiple users, different DOs will product their own $\text{Address}_{\text{DO}}$ and Sign_{DO}, by which they distinguish each other and generate different smart contracts. The DO and CSP need to pay a certain deposit to ensure the smooth completion of subsequent transactions

(2) The DO runs the algorithm KeyGen and outputs a public–private key pair (pk, sk). The algorithm TagGen is run to output the block signatures ψ and the block indices ζ

(3) The DO uploads $\{F, \psi, \zeta\}$ to the CSP, generates T_1, and deploys it on the blockchain. The storage contract T_1 contains basic file information (name, hash, and upload time) and transaction-related information (storage fee, DO's address, CSP's address, and DO's signature). This contract can ensure that the DO must pay the CSP in time if the latter stores all of the files of the former and the data satisfy the proof of verification

(4) When the CSP receives $\{F, \psi, \zeta\}$, it generates T_2 and deploys it on the blockchain. The compensation contract T_2 contains basic file information (name, hash, and receive time) and transaction-related information (compensation fee, DO's address, CSP's address, and CSP's signature). This contract can ensure that the CSP pays/compensates the DO in time if it does not store its entire data such that integrity verification fails

(5) The CSP periodically runs the algorithm ProofVerify to generate the proof $\mathscr{P}$ according to the hash value of the previous block h and the corresponding timestamp ω of the latest block generated. It generates T_3 and deploys it on the blockchain. The verification

contract T_3 contains basic file information (name, hash value, and proof), the related contracts (T_1 and T_2), and transaction-related information (verification fee, CSP's address, verifier's address, and CSP's signature). To obtain the reward, the verifier executes contract T_3 and chooses to activate T_1 or T_2 according to the result

3.4. Brief Summary. We have developed a post-quantum privacy-preserving provable data possession scheme based on smart contracts that not only meets the essential requirements of correctness and security but also has the following characteristics.

(1) Post quantum: the scheme uses the linear homomorphic signature on lattices based on the preimage sampling algorithm. Its security depends on the ISIS problem on lattices and provides a strong proof of security based on the worst-case hardness so that it can meet the requirements of protecting against the quantum attacks

(2) Public verification: the proof verification algorithm of the scheme is public and does not require the use of a private key. Any third-party auditor can obtain a public conclusion on whether the CSP completely stores the DO's data, which liberates users from the arduous task of verification

(3) Privacy preservation: the random masking technology introduces random variables to ProofGen. The adversary cannot obtain the user's original data by solving the linear equations composed of different proofs, which satisfies the demand for privacy preservation

(4) Noninteractive: the scheme uses the time-varying hash value of the previous block to generate the challenge set so that the parties do not need to interact in the traditional "challenge–response" phase. On the one hand, the DO and CSP do not need to stay online all the time, which makes the operation of the scheme more flexible, and on the other hand, it can reduce the cost of transactions due to interactions in the blockchain

(5) Fair payment: the DO, CSP, and verifier trade according to the agreed smart contract. The smart contract is immutable and automatically executed such that fair payment can be guaranteed

4. Proof of Security

4.1. Correctness

Theorem 12. *Our post-quantum privacy-preserving PDP protocol proposed in Section 3.3 is correct with an overwhelming probability.*

Proof. Assume a file $F = \{m_i\}_{1\le i\le k}$, block signatures $\sigma_i \longleftarrow \text{SamplePre}(\Lambda_1 \cap \Lambda_2, T, t_i, v)$, a challenge set $Q = (c_1, c_2, \cdots, c_k)$, and the proof $\mu = s + \sum_{i=1}^{k} r_i c_i m_i$, $\sigma = s + \sum_{i=1}^{k} r_i c_i \sigma_i$, and $\phi = As + \sum_{i=1}^{k} r_i c_i \alpha_i$. We check the three conditions separately in ProofVerify.

(a) By Theorem 4, we have $\|\tilde{T}\| \le O(p \cdot \sqrt{m \log q})$; therefore, $v = p \cdot \sqrt{m \log q} \cdot \log m \ge \|\tilde{T}\| \cdot \sqrt{\log m} \ge \|\tilde{T}\| \cdot \omega(\sqrt{\log m})$. By Lemma 8, we have $\|\sigma_i\| \le v \cdot \sqrt{m}$. Since $\sigma = s + \sum_{i=1}^{k} r_i c_i \sigma_i$, $s \in \mathbb{Z}_2^m$, $r_i \in \{0, 1\}$, and $c_i \in \{0, 1\}$, we have that with overwhelming probability

$$\|\sigma\| \le \|s\| + \left\| \sum_{i=1}^{k} r_i c_i \sigma_i \right\| \le \sqrt{m} + k \max_{1\le i\le k} \|\sigma_i\| \le (vk + 1)\sqrt{m} \tag{7}$$

(b) Since $\sigma_i \in p\Lambda_2 + t_i$ and $t_i \bmod p = m_i$, we have $\sigma_i = t_i = m_i (\text{mod } p)$. Thus, the following equation holds:

$$\sigma = s + \sum_{i=1}^{k} r_i c_i \sigma_i = s + \sum_{i=1}^{k} r_i c_i t_i = s + \sum_{i=1}^{k} r_i c_i m_i = \mu (\text{mod } p) \tag{8}$$

(c) We have $\sigma_i \in p\Lambda_2 + t_i$ and $\Lambda_2 = \Lambda_q^{\perp}(A)$. Since $At_i \bmod q = \alpha_i$, we have $A\sigma_i = At_i = \alpha_i (\text{mod } q)$. Thus, the following equation holds:

$$\begin{aligned} A\sigma &= A \cdot \left(s + \sum_{i=1}^{k} r_i c_i \sigma_i \right) = As + \sum_{i=1}^{k} r_i c_i (At_i) \\ &= As + \sum_{i=1}^{k} r_i c_i \alpha_i = \phi (\text{mod } q) \end{aligned} \tag{9}$$

□

4.2. Soundness. We define the security of the scheme by formally describing a series of games between a challenger $\mathscr{C}$ and an adversary $\mathscr{A}$.

Setup: $\mathscr{C}$ runs the algorithm KeyGen to generate the public-private key pair (pk, sk), reserves sk for responding to the $\mathscr{A}$'s query, and then sends pk to $\mathscr{A}$.

Queries: $\mathscr{A}$ can execute adaptive signature queries by interacting with $\mathscr{C}$. $\mathscr{A}$ specifies a series of j files $F_j = \{m_{ij}\}_{1\le i\le k}$ to send to $\mathscr{C}$. $\mathscr{C}$ generates the corresponding set of j signatures $\psi_j = \{\sigma_{ij}\}_{1\le i\le k}$ and the set of block indices $\zeta_j = \{\alpha_{ij}\}_{1\le i\le k}$ by running the algorithm TagGen and then sends (ψ_j, ζ_j) to $\mathscr{A}$.

Output: $\mathscr{A}$ outputs the proof based on the results of multiple queries. For a file F with a set of signature $\psi = \{\sigma_i\}_{1\le i\le k}$ and a set of block indices $\zeta = \{\alpha_i\}_{1\le i\le k}$, the proof $\mathscr{P}^* = (\mu^*, \sigma^*, \phi^*, h, \omega)$

is output when the hash value of the previous block is h and the timestamp is ω.

Definition 13. The advantage of an adversary $\mathcal{A}$ in the game is $\mathrm{Adv}_{\mathcal{A}} = \Pr\{\mathrm{ProofVerify}(\mathcal{P}^*, \mathrm{pk}) = \mathrm{SUCCESS}\}$, and if $\mathrm{Adv}_{\mathcal{A}}$ is nonnegligible, we say that $\mathcal{A}$ wins the game.

Definition 14. The post-quantum privacy-preserving provable data possession protocol proposed in Section 3.3 is secure. If there exists an efficient extraction algorithm Extr, such that any adversary $\mathcal{A}$ wins the security game and outputs the proof $\mathcal{P}^*$ of the file F, the probability that Extr can recover F from $\mathcal{P}^*$ (i.e., $\mathrm{Extr}(\mathrm{pk}, \mathcal{P}^*) = F$) is not negligible.

Theorem 15. *If the algorithms used to generate the block signatures and block indices of the files are existentially unforgeable and the $ISIS_{q,m,\beta}$ problem is difficult in case $\beta = 2mkp \cdot \log m\sqrt{\log q} + 2\sqrt{m}$, then, under the random oracle model, the probability that any adversary breaking the security of our scheme passes ProofVerify by using a proof not generated by ProofGen is negligible.*

Proof. We demonstrate Theorem 15 through the interactive analysis of a series of games. The game's restrictions on the opponent $\mathcal{A}$ are gradually tightened. □

Game-0: Game-0 is the first game. It is the security game defined at the beginning of this section.

Game-1: Game-1 is similar to Game-0 with one difference: the challenger $\mathcal{C}$ maintains a list of block signatures that have been responded to during the query phase. If the adversary $\mathcal{A}$ submits a valid signature σ_i that is, however, not in the challenger's signature list, $\mathcal{C}$ aborts and outputs failure.

Game-2: Game-2 is similar to Game-1 with one difference: the challenger $\mathcal{C}$ maintains a list that stores the indices of the data blocks that have been responded to during the query phase. If the adversary $\mathcal{A}$ submits a valid index α_i that is not in the challenger's index list, $\mathcal{C}$ aborts and outputs failure.

Game-3: Game-3 is similar to Game-2 with one difference: the challenger $\mathcal{C}$ maintains a list of all tag queries and responses initiated by the adversary $\mathcal{A}$. If $\mathcal{A}$ submits a proof of successful verification where μ is not equal to $s + \sum_{i=1}^{k} r_i c_i m_i$, σ is not equal to $s + \sum_{i=1}^{k} r_i c_i \sigma_i$, and ϕ is not equal to $As + \sum_{i=1}^{k} r_i c_i \alpha_i$, $\mathcal{C}$ aborts and outputs failure.

Lemma 16. *If there is an algorithm $\mathcal{A}$ that can distinguish Game-1 from Game-0 with a nonnegligible probability, we can construct an algorithm $\mathcal{B}$ that can counteract the existence of GPV signature scheme with a nonnegligible probability.*

Analysis: if the adversary $\mathcal{A}$ fails to output in Game-1, we can forge a valid signature σ_i, which contradicts the existential unforgeability of the GPV signature scheme.

Table 2: Comparison of the characteristics of schemes.

	[15]	[17]	[19]	[28]	[30]	Our scheme
Post quantum	×	×	×	√	√	√
Public verification	√	√	√	√	√	√
Privacy-preserving	×	×	√	×	√	√
Noninteractive	√	×	√	×	×	√
Fair payment	√	×	√	×	×	√

Lemma 17. *If there is an algorithm $\mathcal{A}$ that can distinguish Game-2 from Game-1 with a nonnegligible probability, we can construct an algorithm $\mathcal{B}$ for the collision resistance of the hash function with a nonnegligible probability.*

Analysis: if the adversary $\mathcal{A}$ fails to output in Game-2, we can forge a valid index $\alpha_i^* \neq \alpha_i$, but they are all equal to $H_2(\tau \| i)$, which contradicts the collision resistance of the hash function.

Lemma 18. *If there exists an algorithm $\mathcal{A}$ that can distinguish Game-3 from Game-2 with a nonnegligible probability and $\beta = 2mkp \cdot \log m\sqrt{\log q} + 2\sqrt{m}$, we can construct an algorithm to solve the $ISIS_{q,m,\beta}$ problem.*

Analysis: before analyzing the above, we establish some notation. Suppose that the file that caused the failure is divided into k blocks of equal length, denoted as $F = \{m_i\}_{1 \le i \le k}$. The block signature set $\psi = \{\sigma_i\}$ and the block index set $\zeta = \{\alpha_i\}$ are generated by TagGen. Assuming that $Q = (c_1, c_2, \cdots, c_k)$ is the set of challenges that lead to failed queries, the outputs of the adversary's response are μ^*, σ^*, and ϕ^*, where $\mu^* = s^* + \sum_{i=1}^{k} r_i^* c_i m_i$, $\sigma^* = s^* + \sum_{i=1}^{k} r_i^* c_i \sigma_i$, and $\phi^* = As^* + \sum_{i=1}^{k} r_i^* c_i \alpha_i$. Let the expected responses (generated by an honest prover) be μ, σ, and ϕ. We can ensure by Lemmas 16 and 17 that the block signature σ_i and block index α_i used in the queries and output phases, both generated by the challenger $\mathcal{C}$, are unforgeable. Thus, if $\mu^* = \mu$, there must be $\sigma^* = \sigma$ and $\phi^* = \phi$, and if $\mu^* \neq \mu$, there must be $\sigma^* \neq \sigma$ and $\phi^* \neq \phi$.

We now show that if the adversary fails the challenger with a nonnegligible probability in Game-3, we can construct a simulator $\mathcal{S}$ to solve the ISIS problem on lattices.

The inputs of the simulator $\mathcal{S}$ are the file F and some parameters. Its goal is to output a vector $\sigma^* - \sigma$. The behavior of the emulator is different from that of the challenger in Game-2, including in terms of the algorithm setup, queries, outputs, and the hash:

Setup: when generating the key, create a lattice Λ_1 and its short basis T_1. The other parameters are the same as in Game-2, meaning that the simulator does not know the private key T.

Hash: the simulator writes a random oracle H and maintains a table of queries and responses. When the adversary chooses $\tau \| i$ to initiate a query, it returns $H(\tau \| i)$ if it is in the queried list; otherwise, it returns $H(\tau \| i) = A \cdot \sigma_i \bmod q$, where $\sigma_i \longleftarrow \mathcal{D}_{\mathbb{Z}^n, \nu}$.

Table 3: The relationship between the running time of the scheme and the parameter m ($k = 100$).

	Algorithm's time (s)				
	KeyGen	TagGen	ProofGen	ProofVerify	Total
$m = 32$ ($n = 1$, $q = 2560021$)	0.0021	3.762	0.0081	0.0000314	3.7722314
$m = 64$ ($n = 2$, $q = 2560021$)	0.0063	19.2676	0.0145	0.0000648	19.288465
$m = 128$ ($n = 4$, $q = 2560021$)	0.0313	125.0181	0.0251	0.0001594	125.07466
$m = 256$ ($n = 8$, $q = 2560021$)	0.1863	896.074	0.04867	0.0004037	896.30937

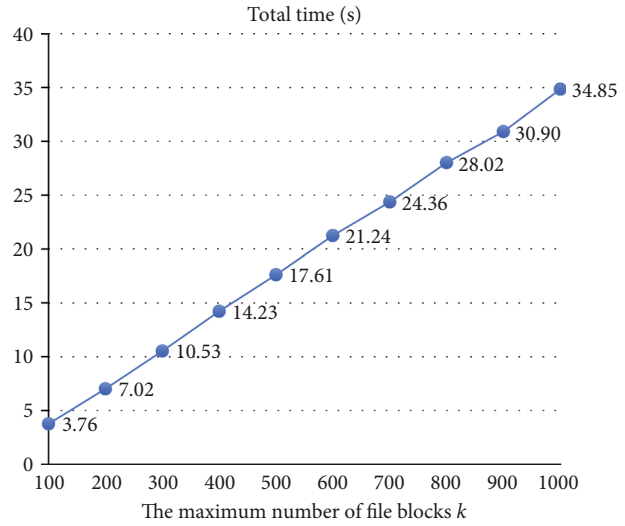

Figure 5: The relationship between the running time of the scheme and the parameter k ($n = 1$, $q = 2560021$, and $m = 32$).

Queries: when asked to sign the file F, the simulator does the following:

(1) It randomly selects $\tau \longleftarrow^R \{0,1\}^m$ as the tag. Since the selection space is large enough, the probability that the simulator selects a tag that has once initiated query $\tau \| i$ to the random oracle H is negligible. If it has been queried, it aborts

(2) Choose t_i such that $t_i \bmod p = m_i$, and output $\sigma_i \longleftarrow \text{SamplePre}(\Lambda_1, T_1, t_i, v)$

(3) Let $H(\tau \| i) = \alpha_i = A \cdot \sigma_i \bmod q$, $1 \le i \le k$

Output: for the challenge set $Q = (c_1, c_2, \cdots, c_k) \in \mathbb{Z}^k$, compute μ^*, σ^*, and ϕ^*, and finally, output the vector $\sigma^* - \sigma$.

We first show that the output of the simulator $\mathcal{S}$ is distributed (up to a negligible statistical distance) as in the real signature scheme. Because the simulator chooses a random tag τ from $\{0,1\}^m$ when signing, the probability that the simulator aborts is negligible. We may therefore assume that the simulator does not abort. Since $v \ge \eta_\varepsilon(\Lambda_1 \cap \Lambda_2)$ for a negligible ε, by Lemma 9, the vectors σ_i chosen in both hash and queries are statistically close to the uniform modulo Λ_2, and thus, the output of $H(\tau \| i) = \sigma_i \bmod \Lambda_2$ is indistinguishable from random. By Theorem 6, σ_i in the real scheme are distributed as $\mathcal{D}_{\Lambda_1 \cap \Lambda_2 + t_i, v}$ such that $t_i \bmod p = m_i$ and $A \cdot t_i \bmod q = \alpha_i$. The simulated σ_i, on the contrary, are distributed as $\mathcal{D}_{\Lambda_1 + t_i, v}$ such that $t_i \bmod p = m_i$, conditioned on $A \cdot \sigma_i \bmod q = \alpha_i$. By a straightforward generalization of Lemma 5.2 in [32], these two distributions are identical. Therefore, sampling σ_i from $\Lambda_1 \cap \Lambda_2$ in the real scheme has the same distribution as sampling from Λ_1 and then modulo Λ_2 in the simulator.

We now show that the output $\sigma^* - \sigma$ of the simulator is a solution to the ISIS problem on lattices. The adversary's proof as response is μ^*, σ^*, and ϕ^*, and the expected responses (generated by an honest prover) are μ, σ, and ϕ. They are different, but all can be successfully verified, so there is $\sigma^* - \sigma = \mu^* - \mu \ne 0 (\bmod p)$. According to

$$\begin{cases} \|\sigma\| \le (vk+1)\sqrt{m}, \\ \sigma = \mu (\bmod p), \\ A \cdot \sigma = \phi (\bmod q), \end{cases} \quad \begin{cases} \|\sigma^*\| \le (vk+1)\sqrt{m}, \\ \sigma^* = \mu^* (\bmod p), \\ A \cdot \sigma^* = \phi^* (\bmod q), \end{cases} \tag{10}$$

$\|\sigma^* - \sigma\| \le \|\sigma^*\| + \|\sigma\| \le 2(vk+1)\sqrt{m} = \beta$. Assuming that $\Delta s = s^* - s$ and $\Delta t = t^* - t$, let $A \cdot \Delta s + \Delta t \cdot \sum_{i=1}^k c_i \cdot \alpha_i = d$. Then, $A \cdot (\sigma^* - \sigma) = \phi^* - \phi = d (\bmod q)$, i.e., $\sigma^* - \sigma \in \Lambda_q^d(A)$. We thus identify the nonzero vector $\sigma^* - \sigma$, which is a solution to the $\text{ISIS}_{q,m,\beta}$ problem on the lattice $\Lambda_q^d(A)$.

Summary. Any proof of successful verification is generated by an honest prover and cannot be forged. The challenger thus always aborts and outputs failure; i.e., any adversary's advantage in Game-3 must be zero. The series of games and Lemmas 16–18 help ensure that any adversary's advantage in Game-0 is negligible. Our proposed post-quantum privacy-preserving provable data possession protocol is thus secure.

4.3. Privacy-Preserving. We hope to ensure that the verifier cannot obtain user data based on the information collected during the verification process through the proof of the following theorem.

Theorem 19. *The verifier cannot obtain m_i from the proof $\mathcal{P} = \{\mu, \sigma, \phi, h, \omega\}$ returned by the CSP.*

Proof. According to $\sigma_i \longleftarrow \text{SamplePre}(\Lambda_1 \cap \Lambda_2, T, t_i, v)$ and $\alpha_i = H_2(\tau \| i)$, we cannot obtain any information about m_i from σ and ϕ. Therefore, the proof can be transformed into one whereby the verifier cannot obtain m_i from μ.

Table 4: Comparison of the running time between our scheme and [28, 30].

	Algorithm's time (s)			
	KeyGen	TagGen	ProofGen	ProofVerify
[28]	0.0024	3.6248	0.0061	0.0000177
[30]	0.0024	3.6912	0.2044	0.0000206
Our scheme	0.0022	3.6468	0.0065	0.0000193

We note that because the challenge set $Q = (c_1, c_2, \cdots, c_k)$ is publicly available, if $\mu' = \sum_{i=1}^{k} r_i m_i$ can be obtained by the verifier, then a set of linear equations for m_i can be obtained after multiple challenges and m_i can be obtained by solving them. We now prove that the verifier cannot obtain μ' from μ.

According to $\mu = s + \sum_{i=1}^{k} r_i c_i m_i$, vectors $R = (r_1, \cdots, r_k)$ and s randomly chosen by CSP are unknown to the verifier; thus, μ' cannot be derived from μ. □

5. Performance Evaluation

5.1. Comparative Analysis. This paper proposed a post-quantum privacy-preserving provable data possession scheme based on smart contracts that can verify whether the CSP holds all user data, and it is secure under the random oracle model. We compared our scheme with similar methods along five dimensions: post quantum, public verification, privacy-preserving, noninteractive, and fair payment. Table 2 shows that our scheme has significant advantages over other schemes.

5.2. Experiments. We design a prototype of our scheme to evaluate its performance. Our experiments rely on the NTL Library (version 11.3.2) for matrix operation, lattice reduction, and SHA-256 for the hash algorithm. All experiments were conducted on a laptop running Windows 10 (x64) equipped with 3.20 GHz AMD Ryzen 7 5800H and 16 GB DDR4 RAM.

The experiment takes the running time of the scheme as the evaluation basis, including KeyGen's time, TagGen's time, ProofGen's time, and ProofVerify's time, and mainly focuses on the relationship between the running time and system parameters n, q, m, k. Subsequently, comparisons will be made with two similar schemes [28, 30] and a BLS-based scheme [15], respectively. All results of experiments are representing 30 trials on average.

First, we analyze the relationship between the running time of the scheme and the parameter m. It is the dimension of the lattice, determined by the security parameter n and the prime number q (i.e., $m = \lfloor 6n \log q \rfloor$). It is also the length of each data block and signature, and the scheme's security increases as m increases. We set the number of file blocks $k = 100$; the experimental results are shown in Table 3. The running time of the scheme increases rapidly with the increase of m, and most of the time is spent on the algorithm TagGen. This is because TagGen needs to call the preimage sampling algorithm SamplePre. When the dimension of the lattice m increases, the computational complexity of performing lattice transformation and solving linear equations is much higher than other algorithms in the scheme.

Table 5: Comparison of the running time between our scheme and [15].

	Algorithm's time (s)			
	KeyGen	TagGen	ProofGen	ProofVerify
[15]	0.00002	0.4201	0.1029	0.0035
Our scheme	0.0313	125.0181	0.0251	0.00016

Second, we analyze the relationship between the running time of the scheme and the parameter k. The experimental results are shown in Figure 5.We set the system parameters $n = 1$, $q = 2560021$, and $m = 32$. The running time of the scheme increases linearly with the number of file blocks k. The reason is that the running time is mainly consumed in the TagGen, and the number of signatures is determined by k.

Next, this scheme is compared with two similar schemes [28, 30]. The main difference between the three schemes is in ProofGen and ProofVerify. [28] uses the linear combination of data blocks and block tags as proof, namely, $\mu = \sum_{i \in I} v_i m_i$ and $\sigma = \sum_{i \in I} v_i \sigma_i$; [30] randomly selects vectors r_1 and r_2, uses lattice theory to construct a random vector r satisfying

$$\begin{cases} r \bmod p = r_1, \\ A^T r \bmod q = r_2, \end{cases} \tag{11}$$

and then uses r_1 and r_2 to hide the evidence

$$\begin{cases} \mu = r_1 + \mu', \\ \sigma = r + \sigma'. \end{cases} \tag{12}$$

As shown in Table 4, our scheme does not need to construct random vectors by solving equations, and the efficiency is significantly better than [30] in ProofGen and ProofVerify; due to the use of random masking technology, the running time of our scheme is comparable to that of [28] which has a slight increase but can provide privacy preservation with higher security.

Finally, this scheme is compared with the BLS-based PDP scheme [15]. To achieve the same signature length of 160 bits as [15], we set $k = 100$, $n = 5$, $q = 2560021$, and $m = 160$. As shown in Table 5, although the efficiency of KeyGen and TagGen in our scheme is not as good as that of the scheme [15], ProofGen and ProofVerify take less time. The first two algorithms are completed before uploading data and only need to be run once. In comparison, the latter two algorithms need to be run for each verification, which is more important for the entire integrity verification, so our scheme verification is more efficient.

6. Conclusion

This paper designed a post-quantum privacy-preserving PDP protocol and used it to propose a scheme to this end based on smart contracts. The proposed scheme has the characteristics of being post quantum and is capable of public verification and privacy preservation. The noninteractive nature of the protocol can reduce the cost of blockchain transactions, and the smart contract with the deposit mechanism can ensure fair payments to all parties. In addition, the scheme's efficiency is reflected in the fact that it mainly uses linear operations, thus avoiding a large number of modular exponential and bilinear pairing operations in the traditional method that can significantly reduce the amount of calculation and improve the efficiency of verification. The main bottleneck of the current lattice-based PDP scheme is that the signature process is slow, and the key and signature in the proposed scheme are long. Future research in the area should focus on ways to solve these problems.

Data Availability

The data used to support the findings of this study are included in the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This research is funded by grants from the National Key Research and Development Program of China (No. 2020YFB1805403) and Foundation of Guizhou Provincial Key Laboratory of Public Big Data (No. 2017BDKFJJ015, No. 2018BDKFJJ008, No. 2018BDKFJJ020, and No. 2018BDKFJJ021).

References

[1] "Wiki, Cloud storage," 2022, http://en.wikipedia.org/wiki/Cloud_storage.

[2] S. Mahalle and R. Jaiswal, "Cloud computing security: a survey," *International Journal of Computer Applications*, vol. 115, no. 6, pp. 21–25, 2015.

[3] "Google loses data as lightning strikes," 2015, https://www.bbc.com/news/technology-33989384.

[4] "Tencent cloud user claims $1.6 million compensation for data loss," 2018, https://technode.com/2018/08/06/tencent-cloud-user-claims-1–6-million-compensation-for-data-loss.

[5] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrusted stores," in *Proceedings of the 14thACM Conference On Computer And Communications Security*, pp. 598–609, Alexandria, Virginia, USA, 2007.

[6] 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.

[7] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 514–532, Springer, Berlin, Heidelberg, 2001.

[8] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: multiple-replica provable data possession," in *The 28th International Conference On Distributed Computing Systems*, pp. 411–420, Beijing, China, 2008.

[9] 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*, pp. 1–9, San Diego, CA, USA, 2010.

[10] Y. X. Zha, S. S. Luo, J. C. Bian, and W. Li, "Multiuser and multiple-replica provable data possession scheme based on multi-branch authentication tree," *Journal on Communications*, vol. 36, no. 11, pp. 80–91, 2015.

[11] J. Liu, K. Huang, H. Rong, and M. Xian, "Privacy-preserving public auditing for regenerating-code-based cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 7, pp. 1513–1528, 2015.

[12] 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.

[13] J. Wu, Y. Li, T. Wang, and Y. Ding, "CPDA: a confidentiality-preserving deduplication cloud storage with public cloud auditing," *IEEE Access*, vol. 7, pp. 160482–160497, 2019.

[14] C. Gritti, W. Susilo, and T. Plantard, "Efficient dynamic provable data possession with public verifiability and data privacy," in *Australasian Conference on Information Security and Privacy*, pp. 395–412, Springer, Cham, 2015.

[15] Z. Li, Y. Xin, D. Zhao, and Y. Yang, "A noninteractive multireplica provable data possession scheme based on smart contract," *Security and Communication Networks*, vol. 2022, Article ID 6268449, 14 pages, 2022.

[16] Y. Zhang, R. H. Deng, X. Liu, and D. Zheng, "Blockchain based efficient and robust fair payment for outsourcing services in cloud computing," *Information Sciences*, vol. 462, pp. 262–277, 2018.

[17] R. Chen, Y. Li, Y. Yu, H. Li, X. Chen, and W. Susilo, "Blockchain-based dynamic provable data possession for smart cities," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4143–4154, 2020.

[18] D. Chen, H. Yuan, S. Hu, Q. Wang, and C. Wang, "BOSSA: a decentralized system for proofs of data retrievability and replication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 4, pp. 786–798, 2021.

[19] H. Wang, H. Qin, M. Zhao, X. Wei, H. Shen, and W. Susilo, "Blockchain-based fair payment smart contract for public cloud storage auditing," *Information Sciences*, vol. 519, pp. 348–362, 2020.

[20] Y. Ming and W. Shi, "Efficient privacy-preserving certificateless provable data possession scheme for cloud storage," *IEEE Access*, vol. 7, pp. 122091–122105, 2019.

[21] J. Zhao, C. Xu, and K. Chen, "Secure and efficient privacy-preserving identity-based batch public auditing with proxy processing," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 13, no. 2, pp. 1043–1063, 2019.

[22] C. Gritti, R. Chen, W. Susilo, and T. Plantard, "Dynamic provable data possession protocols with public verifiability and data privacy," in *International Conference on Information*

Security Practice and Experience, pp. 485–505, Springer, Cham, 2017.

[23] W. Tong, B. Jiang, F. Xu, Q. Li, and S. Zhong, "Privacy-preserving data integrity verification in mobile edge computing," in *2019 IEEE 39th international conference on distributed computing systems (ICDCS)*, pp. 1007–1018, Dallas, TX, USA, 2019.

[24] G. Alagic, J. Alperin-Sheriff, D. Apon et al., *Status report on the second round of the NIST post-quantum cryptography standardization process*, US Department of Commerce, NIST, 2020.

[25] D. Micciancio and O. Regev, "Lattice-based cryptography," in *Post-Quantum Cryptography*, pp. 147–191, Springer, Berlin, Heidelberg, 2009.

[26] X. Zhang and C. Xu, "Efficient identity-based public auditing scheme for cloud storage from lattice assumption," in *2014 Ieee 17th International Conference On Computational Science And Engineering*, pp. 1819–1826, Chengdu, China, 2014.

[27] Y. Yang, Y. Sun, Q. Huang, W. Yin, and F. Chen, "RLWE-based ID-DIA protocols for cloud storage," *IEEE Access*, vol. 7, pp. 55732–55743, 2019.

[28] S. Tan, H. Li, C. Zhikun, and J. Yan, "A method of provable data integrity based on lattice in cloud storage," *Journal of Computer Research and Development*, vol. 52, no. 8, pp. 1862–1872, 2015.

[29] D. Boneh and D. M. Freeman, "Homomorphic signatures for polynomial functions," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 149–168, Springer, Berlin, Heidelberg, 2011.

[30] Y. Tan, W. Fan, and J. Wang, "Privacy-preserving cloud data integrity verification scheme," *Journal of Chinese Computer Systems*, vol. 38, no. 12, pp. 2736–2740, 2017.

[31] D. Micciancio and S. Goldwasser, *Complexity of Lattice Problems: A Cryptographic Perspective*, Springer Science & Business Media, 2002.

[32] J. Alwen and C. Peikert, "Generating shorter bases for hard random lattices," *Theory of Computing Systems*, vol. 48, no. 3, pp. 535–553, 2011.

[33] C. Gentry, C. Peikert, and V. Vaikuntanathan, *How to use a short basis: trapdoors for hard lattices and new cryptographic constructions*, 2008.

[34] D. Micciancio and O. Regev, "Worst-case to average-case reductions based on Gaussian measures," *SIAM Journal on Computing*, vol. 37, no. 1, pp. 267–302, 2007.

[35] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized Business Review*, article 21260, 2008.

[36] N. Szabo, "Formalizing and Securing Relationships on Public Networks," *First Monday*, vol. 2, no. 9, 1997.

[37] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for boardroom voting with maximum voter privacy," in *International Conference on Financial Cryptography and Data Security*, pp. 357–375, Springer, Cham, 2017.

[38] V. Gatteschi, F. Lamberti, C. Demartini, C. Pranteda, and V. Santamaría, "Blockchain and smart contracts for insurance: is the technology mature enough?," *Future Internet*, vol. 10, no. 2, p. 20, 2018.