Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 3688164, 12 pages
https://doi.org/10.1155/2022/3688164

Research Article

A Decentralized Public Auditing Scheme for Secure Cloud Storage Based on Blockchain

Jiannan Chen,[1] **Ying Wang**,[2] **Zhaohui Huang**,[1] **Conghao Ruan**,[2] **and Chunqiang Hu**[2]

[1]*Army Medical University, Chongging 400038, China*
[2]*School of Big Data & Software Engineering, Chongqing University, Chongqing 400044, China*

Correspondence should be addressed to Zhaohui Huang; hzhxa@live.cn

Received 13 May 2022; Revised 11 September 2022; Accepted 15 September 2022; Published 14 October 2022

Academic Editor: A.H. Alamoodi

The cloud storage service has brought great convenience to the customer, which can save massive storage and computation resources via outsourcing the data to cloud service provider (CSP). However, the security issues are the biggest challenge such as data integrity. The user can verify the integrity of outsourced data through a remote data auditing solution without retrieving original data from cloud, however, the auditing procedure has heavy computational overhead, which employs third party auditor (TPA) to conduct auditing task on behalf of users. In this paper, we propose a decentralized public auditing scheme for cloud storage based on blockchain, which removes TPA and increases the number of CSP, the auditing task was assigned to multiple CSPs, and the blockchain technology was used to record the audit process. Meanwhile, the structure of e-voting system is utilized to realize the audit result statistics of multiple CSPs via smart contract, which enhanced the credibility and stability of final auditing result. The theoretical analysis and experimental results demonstrate that proposed scheme is secure and efficient.

1. Introduction

With rapid development of computer science and the emergence of concepts such as Internet of Things (IoT) and big data, cloud computing has been widely applied in both business and personal fields, affecting the way we live and produce [1]. Cloud storage, as one of the contents of cloud computing, has attracted academic and engineering attention due to its advantages of large storage capacity, ready-to-use service, high flexibility, and freedom from platform restrictions [2]. Because these advantages that local storage does not have, more and more enterprises and individuals are migrating their data to cloud storage platforms, where cloud service provider (CSP) provide storage and management services [3–5].

Relying on cloud services, users obtain great convenience, but the security of data outsourcing remains a big concern [6]. For cloud storage, users lose direct control of their data, and all the traditional methods used to verify data integrity cannot be applied to it. Besides, despite its claims of credibility, CSP cannot be fully trusted, it may still hide data corruption from users to preserve their own interests, or deliberately delete data that users rarely access to save storage space [7]. Moreover, there is external adversary trying to steal user data. Therefore, cloud users need a verification scheme to ensure the correctness and integrity of outsourced data.

In order to save bandwidth and communication resources, researchers have proposed several remote data auditing schemes that allow users to verify the integrity of outsourced data without local data backup. At first, private auditing [8, 9] was proposed. The user interacts with CSP to obtain proof of the original data, which verifies the integrity of the data. However, user need to regularly verify data integrity, and frequent interaction with CSP and audit operations can cause significant computing and communication resources consumption. As a consequence, researchers introduced TPA to implement public auditing, which enables users to assign auditing tasks to TPA, and users only need to know the auditing results from TPA [6, 7, 10, 11]. Compared with private auditing, public auditing is obviously more economical and practical, so public auditing is more applied in the auditing scheme. Whereas, in most existing public auditing scheme, TPA was considered to be completely trustworthy and will perform every auditing honestly, which also raises security risks. For example, the auditing process of TPA is

untransparent to users, and users can only be notified of audit results. If an irresponsible TPA only tells the user that the audit results are correct in every auditing without doing any actual audit work, the user's data will be at great risk. In addition to this, TPA is a centralized party; it means that TPA is subject to external attacks or internal faults. Once these effects cause TPA system failure, the auditing process will be affected. Even if the system is working properly, TPA may conspire with CSP to cover up data corruption out of self-interest.

To tackle these challenges, we propose a public auditing scheme based on blockchain and e-voting structure in this paper. The main idea is to employ blockchain technology [12] and e-voting to enhance the security of auditing result. E-voting is a decision-making method that uses internet technology to conduct voting, which first proposed by Chaum [13]. It is not limited by region and time, and has the advantage of convenience, rapidly, easy participation, and low cost. E-voting has gained massive attention from various fields. Traditional e-voting protocols usually employ cryptographic tools [14], such as homomorphic encryption and zero-knowledge, to ensure the security of voting. Nonetheless, there is a manager who supervises the whole voting process of existed e-voting protocols; the failure of the manager will lead to the incorrect result of the vote. The blockchain is well suited to solve such problems as it is known for its data security and decentralisation. As a decentralized distribute ledger, the blockchain is constructed in a distribute network consisting of multiple nodes. Each nodes in the network maintain a distributed ledger that contains all the transaction records recognized in the blockchain. Anyone can access the data in the blockchain. Some researchers have proposed schemes with a combination of e-voting and blockchain [15–18]. In addition to supporting e-voting, we record each audit process on the blockchain to achieve the traceability of the auditing process. We also increase the number of CSP. In our scheme, we assign same auditing tasks to multiple CSPs, and count the independent auditing results of CSPs to obtain the final auditing results. The counting process is done through CSP votes, the final statistical work is completed by the smart contract on the blockchain, which can ensure that the statistical results are reliable and verifiable. In general, our contribution in this paper can be summarized as follows:

(i) We propose a public auditing scheme with enhanced reliability, which employs multiple CSPs to implement same auditing task

(ii) To obtain the audit results of outsourcing data, blockchain-based e-voting structure is proposed. The e-voting process is based on the blockchain records and smart contract, which ensures that the auditing records are not tampered with and the audit results statistics are correct

(iii) We propose data sharing scheme to ensure correct data sharing and malicious data sharer detecting

(iv) We prove the security and reliability of proposed scheme through theoretical analysis, we also evaluate the performance through property comparison and experiments

The remainder of this paper is organized as follows. In Section 2, we review related work related to cloud auditing scheme. The background technologies have been introduced in Section 3. The system model, threat model, and design goals are demonstrated in Section 4. Section 5 gives the detailed description of proposed scheme. We further analyze it.

2. Related Work

With the widely circulated of cloud storage service, researchers have put increasing efforts into integrity auditing and proposed many schemes. Juels and Kaliski [8] firstly proposed provable data posession(PDP) model that allow users to remotely verify the integrity of data in semitrusted server. However, their solution is a private auditing scheme and does not support dynamic updates of data. In the same year, Ateniese et al. [9] proposed the model of provable data possession(PDP), which first introduce the concept of public auditing. They aim to allow anyone to audit the integrity of data by utilizing homomorphic verifiable tags (HVTs). In addition, the model used random sampling to generate data proof, which significantly reduce communication consumption while ensuring security. Hereafter, Ateniese et al. proposed a modified scalable PDP [19], this scheme took advantage of symmetric key cryptography to achieve greater efficiency and safety. Compared with original PDP model, [19] supports dynamic data operation, such as append, deletion, and modification. In [20], Shacham and Waters proposed two improvement PoR schemes. The first one is private auditing scheme that adopt pseudorandom functions, the second one is public auditing that based on BLS signature. Compared with the scheme that based RSA signature, the shorter length of BLS signature can effectively reduce communication costs. Since then, many scheme employed BLS signature to save communication computation and achieve batch auditing [10, 21, 22]. Curtmola et al. [23] proposed a MR-PDP model, which allows users to store multiple backups of one file on the server. When some backups are broken, MR-PDP model can recover files quickly. Except for integrity auditing, researchers have done lots of work in dynamic auditing. In order to realize dynamic data operation in cloud auditing, Erway et al. [24] proposed first fully dynamic solution, they employed rank-based authenticated skip list based on PDP model. Sookhak et al. [25] proposed a new technique, called RDA, that achieves minimum communication and computation burden. They also proposed a new data structure: Divided and Conquer Table (DCT) support full dynamic data operation. Tian et al. [11] proposed auditing data structure Dynamic Hash Table (DHT) and migrated the auxiliary information from CSP to TPA. Shen et al. [21] proposed an public auditing protocol with global and sampling blockless verification and batch auditing, in which they constructed a novel dynamic structure.

The concept of TPA is used in many audit programs, TPA was firstly proposed by Wang et al. in [10]. In their scheme, TPA verifies outsourced data on behalf of customers, helping customers save computing and storage resources. Besides, this scheme utilized HVT and Merkle

Hash Tree [26], a well-studied data authentication structure, to achieve dynamic auditing and batch auditing. However, this scheme setting TPA is completely credible, so it cannot deal with the infidelity of TPA. In their subsequent work [27], Wang et al. employed random masking technology on the basis of [10], which could guarantee that TPA cannot derive customers' original data from integrity proofs. Although many cloud data auditing schemes make use of TPA to replace customers for more audit functions, there are some disadvantages that cannot be ignored. First of all, no matter how trustworthy TPA claims to be, customers cannot trust TPA completely. TPA may infer customers data deliberately, collude with CSP to hide the fact that outsourced data has been corrupted out of self-interests. Next, TPA execute all the auditing tasks, once TPA suffers from external attack or internal failure, it will greatly affect customer's service experience. Finally, there is only one TPA available in many schemes, but thousands of customers ask for service. This poses tremendous challenges to TPA's computing and network transmission speed. To address those problems, Armknecht et al. [28] asked for verification of auditor's behaviors, such as the records of auditing process. Zhang et al. [29] proposed a public auditing scheme CPVA, which takes and protracted auditors into consideration. They recorded the time of each auditing operations through blockchain transaction. Yu et al. [30] did not introduce TPA, but proposed a decentralized auditing blockchain (DAB), which used to collect, store proofs, and enhance the reliability and traceability. In [4], Fan et al. proposed a decentralized auditing scheme Dredas, in which TPA was replaced by smart contract on Ethereum.

3. Preliminaries

In this section, we introduced the preliminaries including Bilinear Map, Dynamic Hash Table, Blockchain and Ethereum, and E-voting.

3.1. Bilinear Map. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of large prime order p. Let g be the generator of $\mathbb{G}$. A bilinear map is a map $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ that occupied following properties:

(i) *Bilinear*: for $\forall x, y, z \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}_p$, there is $e(x^a, y^b) = e(x, y)^{ab}$ and $e(x, y, z) = e(x, y) \cdot e(x, z)$

(ii) *Non-degeneracy*: for generator g, there is $e(g, g) \neq 1$

(iii) *Computability*: there exists an efficient and computable algorithm for computing e

3.2. Dynamic Hash Table. Dynamic Hash Table (DHT) is a novel data structure for dynamic data operation, which is proposed in [11]. Figure 1 shows a sample of DHT. Each row of the two-dimensional table records information about one file, including file ID, the version number of each data block in the file, and the latest update time. DO utilizes DHT to generate block tag, A-CSP utilizes DHT to generate information for verification. Further, dynamic block operation and file operation have become much easier with the assistance of DHT.

3.3. Blockchain and Ethereum. Blockchain was first proposed by Nakamoto and Bitcoin in a paper about electronic cash [31]. It is a chained data structure, which is formed by connecting blocks end-to-end. Each block contains an index, a hash pointer to the previous block, a timestamp, its own hash value, and serval transactions data. The existence of hash pointer guarantees that once a block is modified, the hash value of that block will change, and the next block will not be connected to it by the hash pointer, as will all subsequent blocks. If someone wants to modify the data of a block, he or she must modify all blocks from that block. This principle ensures the security of blockchain. In general, the blockchain can be divided into three types: public blockchain, league blockchain, and private blockchain. In public blockchain, anyone can be a node in the blockchain without getting permission, a prime example is bitcoin. In league chain, a predetermined set of nodes maintain the blockchain, such as serval companies work for the same purpose. In private chain, the blockchain is managed by centralized organization.

Ethereum is an open source blockchain platform with smart contracts. Smart contract is a piece of code recorded on the blockchain, which means that the logic of written code is automatically executed as long as the conditions are met. Except for regular blockchain user account, Ethereum also has smart contract account that controlled by smart contract code on the blockchain. Blockchain user can invoke a smart contract by interacting with the account.

3.4. E-Voting. E-voting is an efficient and cost-saving way for conducting a voting process, which allows user to conduct voting through electronic devices, such as cell phone or computer. To ensure the integrity of the results, e-voting needs an authority to conduct counting and publishing. A complete e-voting system needs to satisfy several principles and requirements [32], but this scheme employs a simplified version.

4. Problem Statement

4.1. System Model. The decentralized auditing architecture of proposed scheme is shown in Figure 2 in previous work [33]. There are three entities: data owner (DO), data user (DU), and CSP. To make it easier to describe data sharing, we will discuss DO and DU separately. In practice, DO and DU can be the same person.

(i) *DO*: has limited computing and storage resources, it outsources large data files to CSP and authorizes other CSPs to verify the integrity of data at regular intervals

(ii) *DU*: acquires the data outsourced in CSP. Besides, for convenience or cost saving, DU will share the data with others. A single piece of data may circulate among many individuals

(iii) *CSP*: provides storage and management services for DO while ensuring data integrity. From DO's perspective, CSP can be divided into two categories by function: S-CSP is responsible for storing users' data and providing data proof for auditing requests,

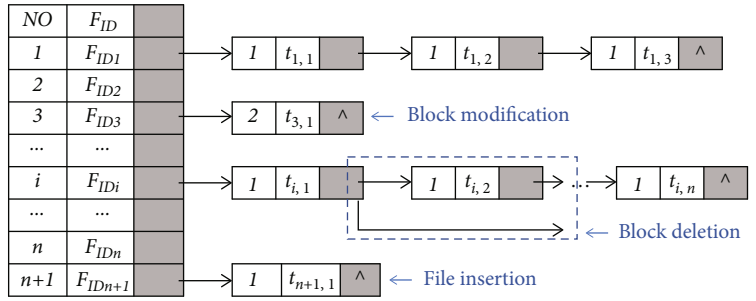

Figure 1: DHT.

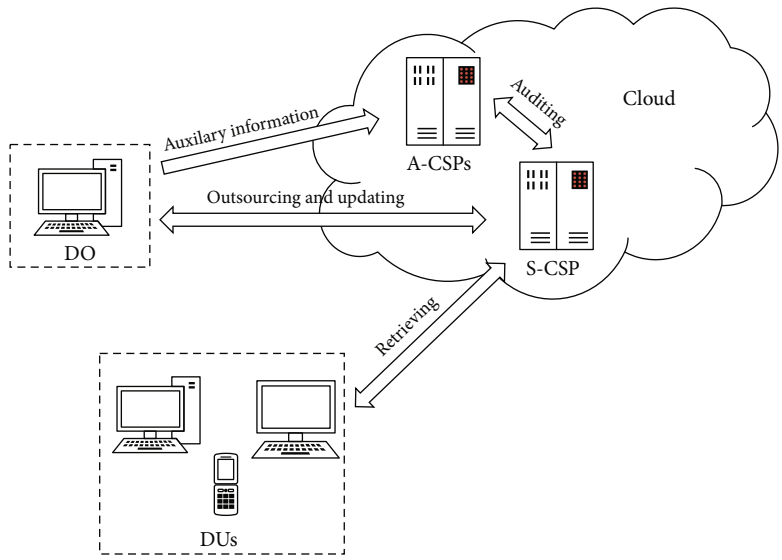

Figure 2: System Model.

A-CSPs are responsible for implementing regular auditing of the data on S-CSP. For a single DO, one CSP performs the function of S-CSP, while the other CSPs perform auditing task together as A-CSP

In proposed scheme, all DO, DU, and CSPs are blockchain user. A-CSP sends auditing request to S-CSP, then S-CSP generates data proof and sends it back to A-CSP. The information exchanges of A-CSP and S-CSP are stored on the blockchain in the form of transaction, and can be accessed by all blockchain users. Therefore, we can realize that all A-CSPs perform the same audit task and get the audits result independently. After that, A-CSPs send auditing results to smart contract for counting and broadcasting. Note that we do not consider the data privacy issues for this topic in cloud storage auditing is orthogonal to what we study in this paper.

4.2. Threat Model. In our scheme, we assume CSP is semi-trusted. For example, CSP performs store and audit reliably, but S-CSP may deliberately conceal data corruption from DO. A-CSP may be compromised, that is to say, A-CSP may collude with S-CSP to give correct auditing results on corrupted data out of self-interest. Besides, there are also security issues because of the introduction of e-voting. More specifically, the following attacks may exist in our scheme:

(i) *Collusion attack*. The CSP may collude to modify the auditing results, so the fact of some data being corrupted would be covered up

(ii) *Forge attack*. The S-CSP may forge outsourced data and corresponding block tag to pass verification

(iii) *Modification attack*. The S-CSP may ask A-CSP to modify historical auditing records for its own reputation

(iv) *Counterfeiting attack*. During voting process, there may be some malicious parties who cast fake votes

4.3. Design Goal. In order to ensure the safety and efficiency of the scheme, we designed to achieve the goals as follows:

(i) *Public auditing*. Anyone (except for the entities in our scheme) is able to verify the integrity and correctness of data store in cloud server

(ii) *Safe storage*. Once outsourced data are corrupted, the auditing results of the data will be false

(iii) *Decentralized auditing*. Multiple A-CSPs audit the same data, and the auditing results do not interfere with each other

(iv) *Blockless verification*. There is no need to retrieve original data for verification

(v) *Traceability*. Every auditing process of every A-CSP can be acquired and validated

(vi) *Data sharing*. In the process of DU sharing the data, the malicious modification of the data can be detected

5. The Proposed Scheme

In this section, we present the proposed scheme, which is based on blockchain technology and e-voting structure. The procedure of the proposed scheme consists of four stages are as follows:

(i) *Setup*: DO generates block tag, file tag, and DHT. Then DO uploads tags along with corresponding file to S-CSP, DHT to A-CSPs

(ii) *Dynamic data operation*: after uploading, DO dynamically updates the data on the cloud server, such as appending, deleting, and modification

(iii) *Integrity verification*: A-CSPs audit the data stored in cloud server, and send respective auditing results to smart contract account, which makes statistics and obtains the final auditing results. Figure 3 shows the process of voting

(iv) *Data sharing*: DUs share data and maintain data integrity during data sharing. If a malicious nodes tampers with and shares the wrong data, subsequent nodes can determine who modified data

5.1. Setup. A DO, multiple DUs, and multiple CSPs are included in our scheme. Before starting, we assumed the file F has been processed (such as encryption) and is divided into n blocks: $F = \{m_1, m_2, \cdots, m_n\}$, $m_i \in Z_p$, and p is a large prime. $\mathbb{G}$ and $\mathbb{G}_T$ is two multiplicative cyclic groups of order p, and $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is a bilinear map. Let H be a hash function $\{0,1\}^* \longrightarrow \mathbb{G}$. h is a cryptographic hash function. The setup procedure can be described as follows:

Key Gen: DO generates secret parameters. Firstly, DO randomly choose $\alpha \in Z_p$, $g, u \in \mathbb{G}$, and $\beta = g^\alpha$. And then, DO chooses a random signing key pair $(\mathrm{sk}, \mathrm{pk})$ for signature. Ultimately, DO set the secret key as $\mathrm{SK} = (\mathrm{sk}, \alpha)$, which is kept by DO itself, and the public key as $\mathrm{PK} = (\mathrm{pk}, \beta, g, u)$.

DI Gen: DO generates information about the files that need to be stored in A-CSPs, the choice of DO for S-CSP and A-CSPs is random, but notice that for one DO there is only one S-CSP. After making the choice, DO generates $DI = \{F_{ID}, \Phi = \{v_i, t_i\}_{1 \le i \le n}\}$, where F_{ID} is preallocated unique identity of file F, $\Phi = \{v_i, t_i\}_{1 \le i \le n}$ represents the version number and latest update time of block m_i. Then DO uploads DI to A-CSPs, and A-CSPs will add it to DHT for this DO.

Tag Gen: DO generates block tags and file tags for files to be outsourced. Firstly, for each block in file $F = \{m_1, m_2, \cdots, m_n\}$, DO computes block tag: $\sigma_i = (H(v_i \| t_i) \cdot u^{m_i})^\alpha$, $1 \le i \le n$. The set of block tags are represented as $\theta = \{\sigma_i | 1 \le i \le n\}$. Whereafter, for file F, DO computes file tag based on secret key sk: $T = F_{\mathrm{ID}} \| \mathrm{sig}_{\mathrm{sk}}(F_{\mathrm{ID}})$. Finally, DO uploads $\{F, \theta, T\}$ to S-CSP, and removes them from local database.

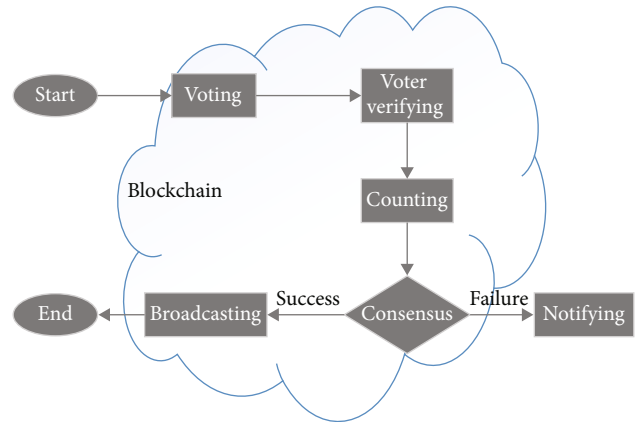

Figure 3: Auditing process.

5.2. Dynamic Data Operation. After uploading all the data, DO may want to perform dynamic operation of the data. In this section, we described block modification and block insertion, the updating operations of file is similar to block. For we store original data and state data separately, data updates also need to be made in two parts. We took advantage of DHT proposed in [11], so our update of DHT is the same as [11].

Block Modification: we assumed that the block m_i need to be replaced by m_i'. Firstly, DO generates new version number and timestamp for m_i', which is (v_i', t_i'). DO computes new block tag $\sigma_i' = (H(v_i' \| t_i') \cdot u^{m_i'})^\alpha$. Then, DO constructs update request $M_{\mathrm{DI}} : \{F_{\mathrm{ID}}, \mathrm{MD}, i, v_i', t_i'\}$, where MD represents modification, and DO sends it to A-CSPs. Upon receiving R_{DI}, A-CSPs find the i-th node of file F and replaces the node content with $\{v_i', t_i'\}$. Meanwhile, DO constructs $M_F : \{F_{\mathrm{ID}}, \mathrm{MD}, i, m_i, m_i', \sigma_i'\}$ and sends it to S-CSP. After receiving, S-CSP replaces the i-th block m_i of file F with m_i', and the corresponding tag σ_i with σ_i'.

Block Insertion: we assumed that block m_i will be inserted after block m_{i-1}. Same as block modification, DO needs to firstly generate new data information (v_i, t_i) for m_{i-1}, then DO sends insertion request $I_{\mathrm{DI}} : \{F_{\mathrm{ID}}, \mathrm{IS}, i, v_i, t_i\}$ to A-CSPs. Upon receiving it, A-CSPs find $(i\text{-}1)$-th node of file F and inserts a new node after it, the content of new node is $\{v_i, t_i\}$. For data inserting, DO computes block tag $\sigma_i = (H(v_i \| t_i) \cdot u^{m_i})^\alpha$ for m_i, and sends insertion request $M_F : \{F_{\mathrm{ID}}, \mathrm{IS}, i, m_i, \sigma_i\}$ to S-CSP. Upon receiving it, S-CSP insert m_i and σ_i into corresponding sets.

5.3. Integrity Verification. In proposed scheme, the process of auditing is built on the blockchain, and the auditing results are voted with the help of smart contract. We denote A-CSPs blockchain accounts as A_{1-m}, which m is the total number of A-CSP. S-CSP blockchain account as S, smart contract account as SC. The integrity verification procedure can be described as follows:

Challenge: since all A-CSPs participates in the same audit task, we randomly select one form A-CSPs, denoted as A_1, to launch challenge to S. Before launching, A_1 need to verify file tag of target file. A_1 acquire file tag T from S and verifies the correctness of it by DO's public key pk. If the verification failed, A_1 would terminate the auditing and notify DO that the data has been corrupted. If not, A_1 regains file ID. Then, A_1 constructs challenge information $\text{chal} = \{i, s_i, R\}_{i\in I}$, in which I is a subset of $[1, n]$ with c elements, representing the index of blocks to be checked. $s \in [1, c]$ is randomly selected from $\mathbb{Z}_p$. $R = \beta^r$ is a random masking, in which $r \in \mathbb{Z}_p$ is a random element. Finally, as shown in Figure 4, A_1 initiates a transaction Tx_{A_1} with S, the transaction data is set as chal.

Response: S gets chal from transaction Tx_{A_1}, and computes response information to proof the integrity and correctness of data. First, S computes tag proof $\Theta = \prod_{i\in I}\theta_i^{s_i}$, which is the aggregation of block tag to be checked. For data proof, S computes $M = \sum_{i\in I} s_i \cdot m_i$. After completing, S initiates a transaction Tx_S with SC. As shown in Figure 5, the transaction data is set as $\{\Theta, M, \text{FT}\}$, in which FT is the deadline of voting.

Auditing: the verification of proposed scheme contains proof verification and voting procedure. In order to ensure that smart contract knows the total number of voters and whether the voters are eligible, before the deployment of smart contract, we put the white list containing the address of CSP account into it. The verification phase can be completed as follows:

(i) *Preparation*: upon completion of the transaction, S informs all of A-CSPs to begin voting

(ii) *Verification*: upon receipt of notice, A-CSPs obtain $\{\text{chal}, \Theta, M\}$ from transaction Tx_{A_1} and Tx_S. For data validation, A-CSPs firstly compute $H = \sum_{i\in I} H(v_i \| t_i)^{s_i}$ based on DHT. Eventually, A-CSPs checks the equation $e(\Theta, g)^r = e(H \cdot u^M, R)$. If the equation holds, the data is correct, or else the data has been corrupted. A-CSPs set ballot = TRUE or ballot = FALSE according to the equation, and signs ballot using their private key. Eventually, A-CSPs initiate transaction with SC, respectively, the transaction data is set as $\{\text{ballot}, \text{sig}_{A_i}\}$. We consider A-CSPs' transaction as the vote by it. SC confirms whether the vote is credited to the total by calling Algorithm 1.

(iii) *Counting and broadcasting*: after the polls close, S would call the Algorithm 2 stored in smart contract to obtain final auditing result. SC sends the auditing result to S in the form of transaction Tx_{SC}, as shown in Figure 6. Except for the result, transaction data contains three address list, which are the addresses of A-CSPs whose audit results are true, false, and the addresses of A-CSPs that was not voted for. These three lists can help DO obtain more information than auditing results. For example, DO can better supervise A-CSPs to perform its duties.

Datasharing: for DU, he or she can browse the content of blockchain and obtain outsourced data from S-CSP out of own requirement. When a DU needs to share the data with another DU, denoted as A shares M_A with B. Let t_a implies current time, A computes $h(M_A)$ first, and signs $h(M_A)\|t_a$ to get Sig_A, then generates $h_A = h(h(M_A)\|\text{Sig}_A)$. Finally, A sends $\{M_A, \text{Sig}_A, h_A\}$ to B. After receiving the data from A, B execute Algorithm 3 to verify if Sig_A and M_A is correct, as well as prepare auxiliary information for data sharing. Figure 7 shows the flow of data when it is shared.

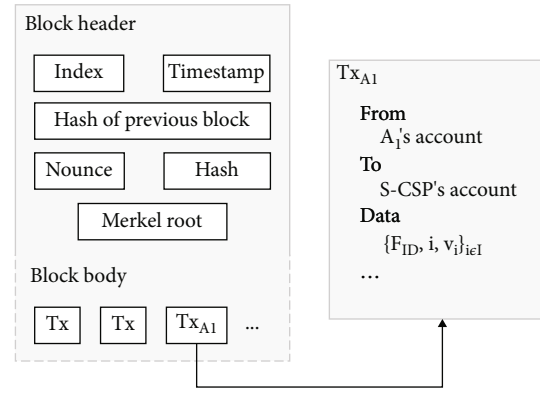

Figure 4: A1's transaction.

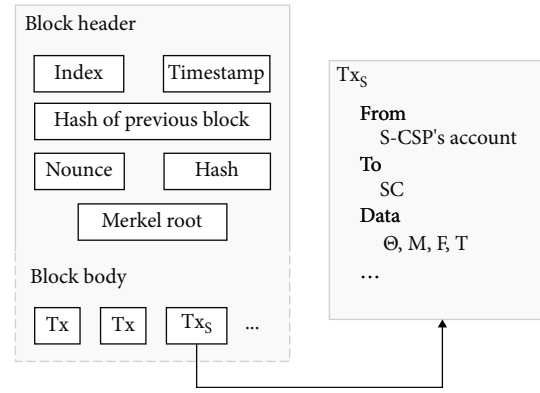

Figure 5: S-CSP's transaction.

6. Security Analysis

In this section, we will prove the security of proposed scheme theoretically.

6.1. Correctness of Verification. The correctness of equation $e(\Theta, g)_r = e(H \cdot u^M, R)$ is elaborated as follows:

$$
\begin{aligned}
e(\Theta, g)^r &= e\left(\prod_{i\in I}\theta_i^{s_i}, g\right)^r = e\left(\prod_{i\in I}(H(\text{vi}\|\text{ti})\cdot u^{m_i})^{\alpha_i\cdot s_i}, g\right)^r \\
&= e\left(\prod_{i\in I}(H(\text{vi}\|\text{ti})\cdot u^{m_i})^{s_i}, g^{\alpha_i\cdot r}\right) = e\left(\prod_{i\in I}(H(\text{vi}\|\text{ti})\cdot u^{m_i})^{s_i}, R\right) \\
&= e\left(\prod_{i\in I}(H(\text{vi}\|\text{ti})^{s_i}\cdot u^{m_i\cdot s_i}), R\right) \\
&= e\left(\prod_{i\in I}H(\text{vi}\|\text{ti})^{s_i}\cdot\prod_{i\in I}u^{m_i\cdot s_i}, R\right) = e\left(H\cdot u^M, R\right).
\end{aligned}
\tag{1}
$$

1: **Input:**Tx_{A_i}
2: **if** Current time is less than the voting deadline **then**
3: **if** The sender of Tx_{A_i} is in the whitelist **then**
4: **if** The sender of Tx_{A_i} has not voted before **then**
5: **if** The signature sig_{A_i} is correct **then**
6: SC record the address of sender in the list according to the content of *ballot*.
7: **else**
8: The vote will not be counted.
9: **end if**
10: **end if**
11: **end if**
12: end if

Algorithm 1: Voting.

1: **Input:**Tx_{S_o}
2: **if** The sender of Tx_{S_o} is the initiator of the vote, which is S-CSP **then**
3: SC generates the addresses list of voters who voted *True*, *False*, and unvoted voters.
4: **if** The number of voters for *True* is greater than 50% of the total number of voters **then**
5: SC sets the voting result to be *True*.
6: **else**
7: SC sets the voting result to be *False*.
8: **end if**
9: **else**
10: Failed vote
11: **end if**

Algorithm 2: Counting.

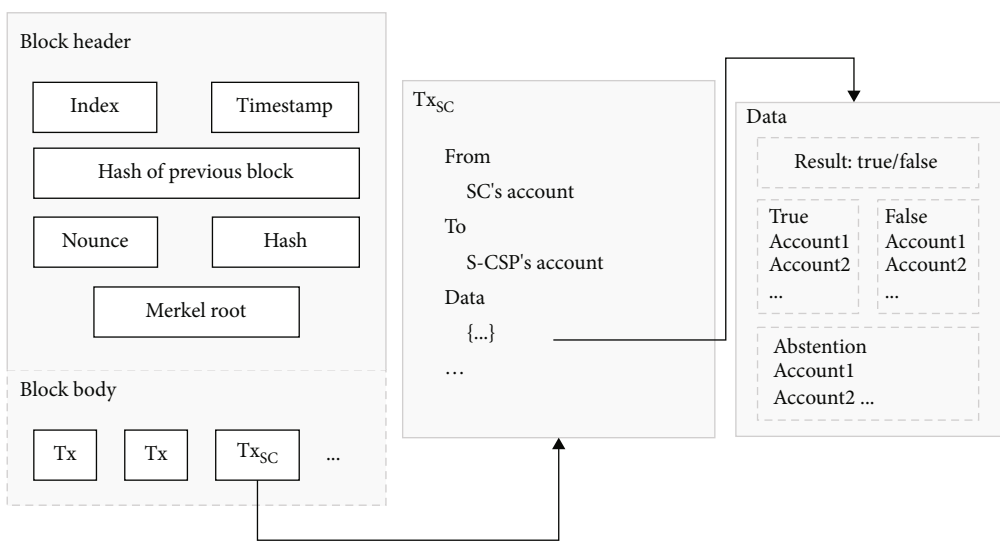

Figure 6: SC's transaction on the blockchain.

6.2. Resisting Collusion Attack. To enhance the reliability of auditing results, we assign same auditing task to multiple A-CSPs, and make use of smart contract on the blockchain to perform auditing results statistics. Only when the auditing result is true for a certain number of A-CSP, will the final auditing result be true. That is, even if S-CSP collude with a few A-CSPs to tamper with auditing results, as long as most of A-CSPs is honest, final auditing result will not be affected. Besides, data on the blockchain is unmodifiable, which means it is impossible to tamper with auditing result by modifying the smart contract.

1: **Input:**$\{M_A, Sig_A, h_A\}$
2: **Output:**$\{M_B, Sig_A, Sig_B, h_B\}$
3: DU B obtains A's public key for validating Sig_A, compares h_A and $h(h(M_A)\|Sig_A)$.
4: **if** Sig_A iscorrect **then**
5: **if** $h_A = h(h(M_A)\|Sig_A)$ **then**
6: It indicates that M_A is correct, let t_b implies current time, B computes $h(M_B)$ and signs $h(M_B)\|t_b$ to get Sig_B, in which M_B represents the data to be sent by B, $M_A = M_B$ when B does not modify M_A. Next, B computes $h_B = h(h_A\|h(M_B)\|Sig_B)$.
7: **else**
8: It indicates that there is a malicious DU that modifies the data, B executes *Algorithm* 4 to find the malicious DU.
9: **end if**
10: **else**
11: Request data again.
12: **end if**
return$\{M_B, Sig_A, Sig_B, h_B\}$

Algorithm 3: Data Sharing.

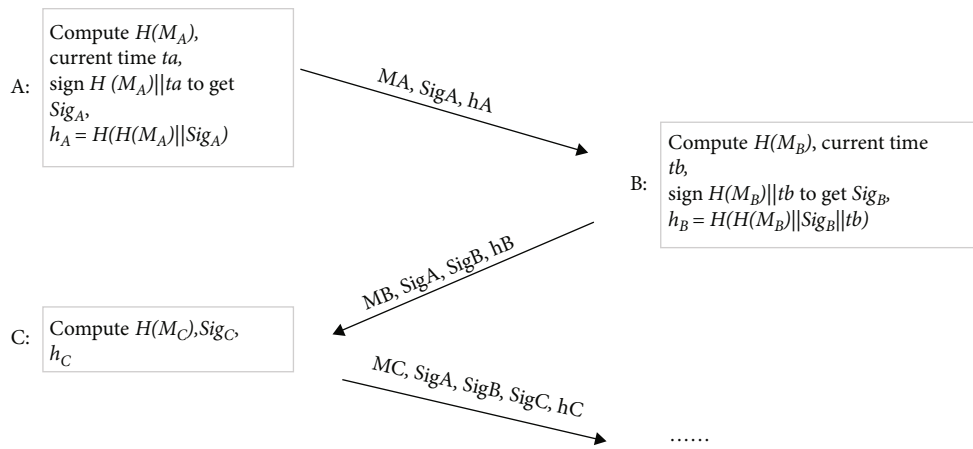

Figure 7: Data sharing process

1: DU D_j verify whether the received data M_{j-1} is correct by checking the equation $h_{j-1} = h(h_{j-2}\|h(M_{j-1})\|Sig_{j-1})$
2: **if** This equation holds **then**
3: D_j accepts the data M_{j-1}.
4: **else**
5: D_j divides DU before D_j into two groups: $D_{1\cdots j/2}, D_{(j/2+1)\cdots(j-1)}$. Then D_j checks if $h_{j/2} = h(h_{j/2-1}\|h(M_{j/2})\|Sig_{j/2})$.
6: **if** The above equation holds **then**
7: The malicious DU exists in $D_{(j/2+1)\cdots(j-1)}$. D_j computes $h_{j/2}$, then verifies if $h_{3j/4} = h(h_{3j/4-1}\|h(M_{3j/4})\|Sig_{3j/4})$.
8: **if** The equation is satisfied **then**
9: The malicious DU belongs to $D_{(3j/4+1)\cdots(j-1)}$, D_j continues to search $D_{(3j/4+1)\cdots(j-1)}$ through binary search until it finds the malicious DU D_i.
10: **end if**
11: **else**
12: The malicious DU exists in $D_{1\cdots j/2}$. D_j continues to search $D_{1\cdots j/2}$ through binary search until it finds the malicious DU D_i.
13: **end if**
14: **end if**

Algorithm 4: Malicious DU detecting

6.3. Detecting Malicious DU. DU shares the data after retrieving it from S-CSP. In the process of data sharing, all the participated DUs want the data to be complete and correct. And even if a DU maliciously tampers with it, that DU can be found. Since the signature and other verification message are contained in shared data, DU can verify the correctness of the data

or find the malicious DU through data validation. We assume that D_i modified the data and send it. Auxiliary message contains $M_i, Sig_1, Sig_2, \cdots, Sig_i, h_i$. $D_j(i < j)$ received modified data and verify it. D_j calls algorithm 4 to verify the data.

6.4. Resistant to Attacks. We validate the attacks mentioned in threat model that can be resisted, the details are as follows:

(i) *Forge Attack*: in the case of data corruption, S-CSP may forge data to pass the verification. But before outsourcing, DO generates block tags and file tag by BLS signature and DO's secret key. According to Wang et al. [10], as long as the CDH problem is hard in bilinear groups, the BLS signature is secure

(ii) *Modification Attack*: in our scheme, all auditing records and auditing results are stored on the blockchain. The unmodifiable nature of the data on the blockchain ensures the security of auditing data

(iii) *Counterfeiting Attack*: we utilize smart contract to conduct auditing results counting and publishing. And before deploying, we put the white list containing the address of CSP account into smart contract. After receiving the vote, smart contract can judge whether the vote comes from qualified A-CSP. In addition, anyone can monitor the implementation of smart contract to ensure the credibility

7. Performance Evaluation

In this section, we will describe the performance evaluation of proposed scheme from the perspective of property comparison and experiments.

The properties comparison between our scheme and other state-of-the-art schemes are shown in Table 1. The letter Y and N indicate that the scheme has this property or not. We can see that the properties of our scheme are relatively complete. Because of the decentralized auditing structure, our scheme is more stable in the face of collusion attack. We can also identify malicious modifiers in the data sharing process.

Table 2 shows the computational cost of DO, DU, and CSP in different phase during auditing. M denotes the multiplication operation on the group, E denotes the exponentiation operation on the group, BP denotes the bilinear pairing operation, H is the general hash function, n is the total number of data blocks in a file, and c denotes the number of blocks to be checked. In our scheme, the computational consumption is mainy generated by TagGen in phase setup, response, and verification in phase integrity verification. For TagGen, DO generates block tags for every blocks, the computation cost is $n(2M + E)$. For response, S-CSP computes tag proof and data proof, thus the computation cost is $(2c - 1)M + cE$. For verification, each A-CSP performs $cM + (c + 2)E + 2\text{BP}$ to verifies the data integrity. Besides, the accounting and publishing of audit results are done by smart contract, that is blockchain network. Therefore, the computational cost of counting and broadcasting is approximately equal to 0.

Table 1: Property comparison with other schemes.

Scheme	Public auditability	Traceability	Collaborative auditability	Data sharing
[21]	Y	N	N	N
[29]	Y	Y	N	N
[30]	Y	Y	N	N
[34]	Y	Y	N	N
Our scheme	Y	Y	Y	Y

Table 2: The computation costs in different phase.

Phase	Computation costs
Key generation	E
TagGen	$n(M + 2E)$
Response	$(2c - 1)M + cE$
Verification	$cM + (c + 2)E + 2\text{BP}$
Announce voting results	0
Data sharing	$(z/2 + 1)H$

Specially, in the data sharing phase, the computation cost increases with the increase of the number of sharers For example, the zth sharers need to perform z general hash functions to verify the correctness of shared data. Because the sharer's signature is superimposed in the order of sharing to get a hash value for data validation, if there is a malicious sharer who modified the data deliberately, the latest sharer can recognize the malicious sharer by binary search, and the time complexity is $O(\log(z))$. As we can see, the computation cost and communication cost in data sharing is very few.

We conduct simulation experiments to validate the efficiency and effectiveness of our scheme. The experiments are performed on a laptop running Windows 7 with a 2.4 GHz Intel Core i7-4500U CPU and 4 GB of memory. We utilizing Pairing-Based-Cryptography(PBC) library version 0.5.14 to implement all the algorithms. And we employ type A pairing parameters, in which the group order is 160-bit. For the data used in experiments, we set the block size as 10 KB, all running time statistics were averaged over 20 trials.

Figure 8 shows the computation time of three phase while total number of blocks are changing. Except for the computational cost of Verification almost stay stable, the computation time of both TagGen and Response grow steadily and linearly as the number of blocks increases. For Verification, we use aggregated data proofs and only one equation to verify them, so the calculation time is stable. The time consumption of TagGen and Response is related to the number of blocks. In these two stages, DO needs to generate block tags one by one, and S-CSP needs to aggregate these together.

Figure 9 shows the proof time of our scheme over two typical ones (i.e., Reference [27] and Reference [11]). From

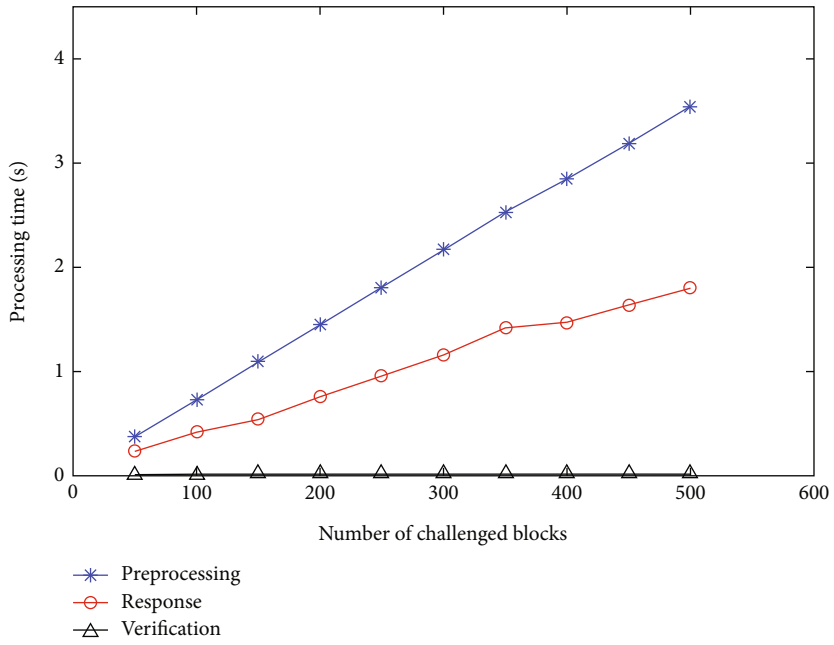

Figure 8: Performance of proposed scheme.

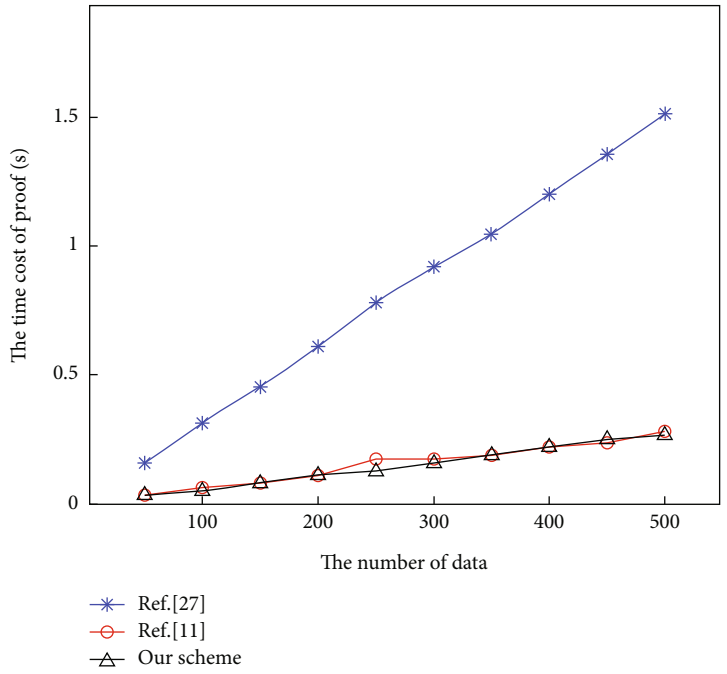

Figure 9: Time cost of proof.

Figure 9, our scheme is better than Reference [27] and Reference [11] in the proof time.

8. Conclusion

In this paper, we proposed a blockchain-based decentralized public auditing scheme. Our scheme employs blockchain and e-voting structure to realize decentralized auditing and collaborative auditing, which improves the stability and reliability of auditing results. In this way, data owners can verify the consistency of the data and quickly find the tampers. We made theoretical analysis and experimental evaluation of the scheme, the results show that proposed scheme meets expected design goals, it is both secure and reliable.

Data Availability

In this paper, we provide the detailed presentation on data in Section VII (Performance Evaluation), meanwhile, we also introduce the procedure of the computational cost analysis. The experiments are performed on a laptop running Windows 7 with a 2.4 GHz Intel Core i7-4500 U CPU and 4 GB of memory. We utilize Pairing-Based-Cryptography(PBC) library version 0.5.14 to implement all the algorithms. And we employ type A pairing parameters, in which the group order is 160-bit. For the data used in experiments, we set the block size as 10 KB, all running time statistics were averaged over 20 trials. The experimental results can be verified the above experimental results in the same running environment.

Disclosure

The previous work [33] was published in International Conference on Wireless Algorithms, Systems, and Applications 2020.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported partially by the Fundamental Research Funds for the Central Universities (No. 2022CDJKYJH015), National Natural Science Foundation of China (No. 62072065), Key Project of Technology Innovation and Application Development of Chongqing (CSTC2019jscx-mbdxX0044), and Overseas Returnees Innovation and Entrepreneurship Support Program of Chongqing (cx2020004).

References

[1] A. Yassine, S. Singh, M. S. Hossain, and G. Muhammad, "IoT big data analytics for smart homes with fog and cloud computing," *Future Generation Computer Systems*, vol. 91, pp. 563–573, 2019.

[2] Z. Liu, C. Hu, R. Li et al., "A privacy-preserving outsourcing computing scheme based on secure trusted environment," *IEEE Transactions on Cloud Computing*, pp. 1–12, 2022.

[3] K. N. Qureshi, F. Bashir, and S. Iqbal, "Cloud computing model for vehicular ad hoc networks," in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, pp. 1–3, Tokyo, Japan, 2018.

[4] K. Fan, Z. Bao, M. Liu, A. V. Vasilakos, and W. Shi, "Dredas: decentralized, reliable and efficient remote outsourced data auditing scheme with blockchain smart contract for industrial IoT," Future Generation Computer Systems, 2020.

[5] S. Cao, G. Zhang, P. Liu, X. Zhang, and F. Neri, "Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain," *Information Sciences*, vol. 485, pp. 427–440, 2019.

[6] R. Kumar and R. Goyal, "On cloud security requirements, threats, vulnerabilities and countermeasures: a survey," *Computer Science Review*, vol. 33, pp. 1–48, 2019.

[7] D. Ardagna, "Cloud and multi-cloud computing: current challenges and future applications," in *2015 IEEE/ACM 7th International Workshop on Principles of Engineering Service-Oriented and Cloud Systems*, pp. 1-2, Florence, Italy, 2015.

[8] A. Juels and B. S. Kaliski Jr., "Pors: proofs of retrievability for large files," in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 584–597, Alexandria, Virginia, USA, 2007.

[9] 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, Alexandria, Virginia, USA, 2007.

[10] 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, 2009.

[11] H. Tian, Y. Chen, C.-C. Chang et al., "Dynamic-hash-table based public auditing for secure cloud storage," *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 701–714, 2017.

[12] Z. Liu, C. Hu, H. Xia, T. Xiang, B. Wang, and J. Chen, "SPDTS: a differential privacy-based blockchain scheme for secure power data trading," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[13] D. L. Chaum, "Untraceable electronic mail, returns address, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[14] Z. Xiong, Z. Cai, C. Hu, D. Takabi, and W. Li, "Towards neural network-based communication system: attack and defense," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–14, 2022.

[15] Z. Zhu, G. Qi, M. Zheng, J. Sun, and Y. Chai, "Blockchain based consensus checking in decentralized cloud storage," *Simulation Modelling Practice and Theory*, vol. 102, p. 101987, 2020.

[16] 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, 2017.

[17] N. Kshetri and J. Voas, "Blockchain-enabled e-voting," *IEEE Software*, vol. 35, no. 4, pp. 95–99, 2018.

[18] K. Garg, P. Saraswat, S. Bisht, S. K. Aggarwal, S. K. Kothuri, and S. Gupta, "A comparitive analysis on e-voting system using blockchain," in *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pp. 1–4, Ghaziabad, India, 2019.

[19] 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, New York, NY, United States, 2008.

[20] H. Shacham and B. Waters, "Compact proofs of retrievability," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 90–107, Springer, 2008.

[21] J. Shen, J. Shen, X. Chen, X. Huang, and W. Susilo, "An efficient public auditing protocol with novel dynamic structure for cloud data," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 10, pp. 2402–2415, 2017.

[22] A. Yang, J. Xu, J. Weng, J. Zhou, and D. S. Wong, "Lightweight and privacy-preserving delegatable proofs of storage with data dynamics in cloud storage," IEEE Transactions on Cloud Computing, 2018.

[23] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "Mr-pdp: multiplereplica provable data possession," in *2008 the 28th*

international conference on distributed computing systems, pp. 411–420, Beijing, China, 2008.

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

[25] M. Sookhak, A. Akhunzada, A. Gani, M. Khurram Khan, and N. B. Anuar, "Towards dynamic remote data auditing in computational clouds," *The Scientific World Journal*, vol. 2014, Article ID 269357, 12 pages, 2014.

[26] R. C. Merkle, "Protocols for public key cryptosystems," in *1980 IEEE Symposium on Security and Privacy*, pp. 122–122, Oakland, CA, USA, 1980.

[27] C. Wang, S. S. 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.

[28] F. Armknecht, J.-M. Bohli, G. O. Karame, Z. Liu, and C. A. Reuter, "Outsourced proofs of retrievability," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 831–843, New York, NY, United States, 2014.

[29] 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. 9, pp. 923–937, 2019.

[30] H. Yu, Z. Yang, and R. O. Sinnott, "Decentralized big data auditing for smart city environments leveraging blockchain technology," *IEEE Access*, vol. 7, pp. 6288–6296, 2018.

[31] S. Nakamoto and A. Bitcoin, "A peer-to-peer electronic cash system," *Bitcoin*, vol. 4, p. 2, 2008, URL: https://bitcoin.org/bitcoin.pdf.

[32] D. A. Gritzalis, "Principles and requirements for a secure e-voting system," *Computers & Security*, vol. 21, no. 6, pp. 539–556, 2002.

[33] Y. Wang, C. Ruan, and C. Hu, "A blockchain-based decentralized public auditing scheme for cloud storage," in *Wireless Algorithms, Systems, and Applications -15th International Conference, WASA 2020*, pp. 482–493, Springer, 2020.

[34] N. Kaaniche and M. Laurent, "Bdua: blockchain-based data usage auditing," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp. 630–637, San Francisco, CA, USA, 2018.