Journal of Engineering, Science and Mathematics

Volume 02, Number 02, Pages 14-35, Year 2023
https://jesm.in/archives/

Research Paper

Artificial Intelligence and Blockchain Based Optimized and Fair Payment Scheme over Cloud Computing System

Renato R. Maaliw III[1], Sheshang Degadwala[2]

[1]Southern Luzon State University, Lucban, Quezon, Philippines

[2]Sigma Institute of Engineering, Engineering Block, Sigma Group of Institutes, Ajwa-Nimeta Road, Bakrol, Vadodara, Gujarat 390019, India

Correspondence should be addressed to Renato R. Maaliw III; rmaaliw@slsu.edu.ph

Handling Editor: Rijwan Khan

An increasing number of data owners (DOs) are opting to move their data to the cloud due to the advent of the cloud storage paradigm. The cloud storage data integrity verification approach is often adopted by the DO in order to guarantee the integrity of the data that it stores in the cloud. In a pay-as-you-go cloud environment, DOs must pay additional costs to a third-party validator (TPA) for carrying out verification procedures, on top of the rates they must pay to cloud service providers. But TPA isn't totally reliable in the integrity verification itself. To address the untrustworthy issue of TPA and achieve service payment fairness, a data integrity verification method that promotes privacy protection and equitable payment is suggested. In order to achieve data location integrity verification and verifiable dynamic data updating, a new kind of data authentication structure called a Merkle hash tree based on hierarchy is first introduced; second, an Interactive Dynamic Data Integrity Proof Mechanism (NIDPDP) is introduced to acknowledge data privacy protection and minimize communication overhead. In order to ensure that all parties really adhere to the regulations that are enforced, smart contracts (SCs) in conjunction with blockchain technology are utilized to accomplish equitable service payment between DO, cloud storage servers (CSS), and TPA. The approach may successfully secure user data privacy, achieve fair payment, and have lower computational and communication overhead, as demonstrated by performance analysis and trials.

Keywords: *Artificial Intelligence, Cloud System, Blockchain, Markle Algorithm, Payment Methods.*

1. Introduction

With the advent of the mobile Internet, Internet of Things, and big data era, the amount of data is exponentially increasing [1-3], and the personal storage capacity cannot meet existing storage needs. Cloud storage is a new concept extended and developed from the concept of cloud computing. Through the functions of cluster application, network technology, or distributed file system, a large number of different types of storage device in the network are integrated and worked together through application software, providing jointly external data storage and business access [4-8]. Users can access files stored in the cloud through the Internet, without any time or location restrictions, and can enjoy high-quality cloud services by paying a certain service fee in a pay-as-you-go manner. Compared with traditional local storage, cloud storage is undoubtedly a more economical choice. However, after users outsource data to efficient cloud services, they no longer have substantial control over the data [9-11]. If the cloud service provider is a semi-honest and curious entity, then it may maliciously delete or tamper with user data in a relationship of economic interests.

To avoid the risk of server (cloudstorage server, CSS) and other users snooping, the concept of data integrity verification [12] was proposed. One of the most intuitive verification methods is that the data owner (DO) downloads all data from CSS and then performs verification locally. However, this only wastes a lot of network transmission and local storage resources. At the same time, in integrity verification, since neither DO nor CSS can guarantee fair and credible results, they are not suitable for performing integrity verification. The literature [13-33] introduces a third-party auditor (TPA) in integrity verification and assigns a TPA with powerful computing power to perform this verification task, so that TPA can provide audit services from an objective and independent perspective. As shown in Figure 1, the TPA-based verification model is mainly composed of three entities: DO, CSS and TPA.

In the verification model, the DO with storage requirements stores local resource data in the server CSS with strong storage and computing capabilities; when the DO performs integrity verification on the data in the cloud, it entrusts a TPA with rich audit experience and capabilities. Execute the verification task; TPA initiates a challenge request to CSS to perform the verification task; CSS generates integrity evidence and returns it to TPA; TPA obtains the verification result through complex calculation and sends the result to DO; DO judges whether the cloud data have been verified according to the returned verification result.

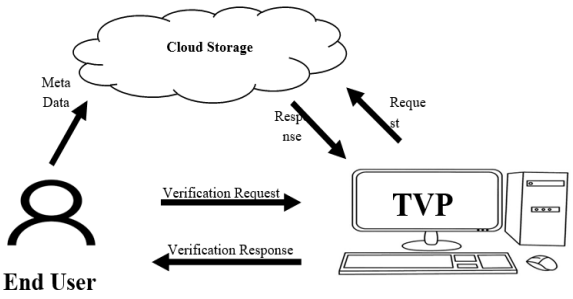

Figure 1: Cloud Storage Data Integrity Verification Model

To support dynamic update verification, Reference [13] made a simple modification to the data ownership proof mechanism (Provable Data Possession, PDP) to support some dynamic data update operations. To fully support the dynamic update operation, Reference [14] proposed a PDP mechanism based on the skip table, which uses the dynamic data structure of the authenticated skip table to

ensure the correct location of the data block, while the data block label ensures the correctness of the content of the data block. However, the existing problem is that the authentication path is too long and a large amount of auxiliary information is needed in the authentication process, which will cause the system to generate a large computational and communication overhead. Reference [15] proposes another public audit scheme that supports dynamic operations, which ensures the accuracy of the location of data blocks by building a dynamic structure of the Merkle hash tree. Compared to the skip table data structure, the Merkle hash tree uses data blocks instead of labels to calculate the hash value of the root node, so it has a simpler data structure. However, in practical scenarios, we need to conduct a specific discussion on the credibility of TPA, that is, TPA is not completely credible and may steal user data privacy.

In order to support the protection of data privacy, based on the homomorphic key random mask technology, Reference [20] proposes to introduce two parameter hidden servers in the audit stage to generate evidence content to ensure that TPA cannot obtain any useful information from returned evidence. The data verification scheme proposed by Reference [23] supports multiuser dynamic multiserver privacy protection, and performs dynamic data verification for multiple users from multiple servers. Combined to provide data privacy protection, Reference [32] proposes a more novel and efficient public verification scheme based on the premise of ensuring data privacy, the dynamic data update structure consists of a bidirectional index information table DLIT and an array of record positions. To locate a data block, the position array can maintain the relationship between the block and its specific position. Among them, the DLIT bidirectional information index table can ensure that other records in the information table will not be changed when inserting and deleting data blocks, so it will have lower system overhead than other schemes. However, the problem in [32] is that in the integrity verification process, after the DO sends an integrity challenge request to the CSS, the CSS returns a corresponding certificate to the TPA to indicate that the outsourced data is stored correctly. Moreover, the TPA may collaborate with the CSP to attack, so it is necessary to ensure the reliability of the TPA verification results.

Due to its decentralization and immutability of data, blockchain has natural advantages in the field of data protection, thus providing a new idea for verifying data integrity. The literature [33-40] proposed a blockchain-based data integrity verification scheme. The distributed storage method enables each participating node on the chain to save a copy of the entire database, which effectively avoids the single-point failure problem of centralized storage. The chain-pointer structure of the blockchain itself can ensure that the data on it cannot be arbitrarily deleted, which is an effective guarantee for data integrity. Reference [35] proposed an ODPDP scheme that delegates frequent auditing tasks to the outside and provides log auditing at the same time to resist any dishonest participants colluding. Reference [36] proposes a SIBAS scheme based on aggregate signatures based on security identity. The Trusted Execution Environment (TEE) acts as a verifier to verify the accuracy of the aggregate signatures and perform integrity verification, which effectively reduces the possibility of data information leakage and

verification results. Untrustworthy and Privacy Protection,

Most existing integrity verification schemes [33-41] are carried out on the premise that the service payment is fair. However, in the actual storage verification, since the DO has paid the corresponding service fee to the CSS and TPA in advance, if the CSS or TPA has deceitful behavior, the fair payment cannot be guaranteed. To solve the untrustworthy TPA problem and realize service payment fairness, this paper proposes a data integrity verification scheme that supports privacy protection and fair payment..

The main contributions of this paper include the following aspects:

i. Introduce a new type of data authentication structure, level-based Merkle hash tree, to realize the integrity verification of data location and support verifiable dynamic update of data.

ii. In order to realize the privacy protection of user data, a non-interactive dynamic data integrity proof mechanism NIDPDP is designed. In the audit stage, the TPA does not need to send a challenge request to the CSS; that is, the challenge interaction process between the TPA and the CSS is cancelled to avoid data

iii. Combined with the consensus and immutability of the blockchain, on the NIDPDP verification mechanism, a data integrity verification model is proposed that supports privacy protection and fair payment. The smart contract (SC) realizes the fair payment content of the verification scheme by punishing the dishonest behaviour of CSS and TPA, and ensures the security, reliability, and fairness of the model.

2. Motivation to investigate

2.1. Overview of the Verification Scheme

Cloud storage data integrity verification refers to a mechanism by which CSS can prove that it correctly stores user data and that user data is kept intact. The verification mechanism generally includes five algorithms, namely key generation, label generation, challenge request, evidence generation, and evidence verification. The scheme that supports dynamic verification also includes two algorithms, namely execution update and update verification.

Key Generation: DO run a probabilistic algorithm to generate public key and private key, and publish public key information.

Label generation: DO divides the data file into blocks and calculates the label corresponding to each data block as the authentication metadata. Then, upload the original file and the set of tags composed of all data blocks to CSS. Finally, delete the local file and tag set.

Challenge request: TPA accepts the verification task entrusted by DO and verifies the file data block by sampling. A random challenge number is generated for the data blocks that need to be verified, the random challenge number and the block index constitute a challenge set, and the TPA challenges the CSS.

Evidence Generation: After CSS accepts the challenge of TPA, it calculates the integrity evidence of this request according to the challenge set and returns it to the verifier.

Evidence verification: After TPA receives the integrity evidence; it verifies the evidence through calculation, and returns the verification result to DO.

Perform the update: The CSS updates the stored data according to the update request and returns the updated evidence to the verifier through the evidence generation algorithm.

Update verification: The TPA receives the updated evidence, executes the update verification algorithm, and returns the update verification result to the DO.

2.2. Attack Model

When the TPA-based integrity verification scheme is adopted, it may cause a leakage of user data privacy. After the DO delegates the integrity verification task to the TPA, if the TPA is not honest and trustworthy, then it may repeatedly verify the data blocks in the same position, that is, the challenge request sent each time is the same. In this way, after accumulating challenge requests for a certain number of times, TPA can obtain a set of equations and then obtain the file information content through the Gaussian elimination method, so that users are exposed to the risk of privacy leakage. The specific attack process is as follows: TPA repeatedly verifies the integrity of the data block at the position s1,s2,..., scs1sisc. After c challenge requests, TPA can get by the system of equations:

$$\begin{cases} v_{s_1}^{(1)} m_{s_1} + v_{s_2}^{(1)} m_{s_2} + \cdots + v_{s_c}^{(1)} m_{s_c} = \mu^{(1)}, \\ v_{s_1}^{(2)} m_{s_1} + v_{s_2}^{(2)} m_{s_2} + \cdots + v_{s_c}^{(2)} m_{s_c} = \mu^{(2)}, \\ \vdots \\ v_{s_1}^{(c)} m_{s_1} + v_{s_2}^{(c)} m_{s_2} + \cdots + v_{s_c}^{(c)} m_{s_c} = \mu^{(c)}, \end{cases} \quad (1)$$

Among them, μ(j) is the evidence element returned by CSS in the j-th integrity verification process, 1jc.

As long as the coefficient determinant in the equation system is not 0, TPA can calculate the value of $\{m_{s_1}, m_{s_2}, \ldots\ldots m_{s_c}\}$ by solving the linear equation system, resulting in privacy leakage of user data.

2.3. Support Privacy Protection Scheme

To verify the protection of data privacy, the literature [20-22] adopts homomorphic key random mask technology. The core idea of this technology is to introduce two parameters in the audit stage to hide the server to generate the evidence content, so as to ensure that the TPA cannot obtain the evidence from the returned evidence to obtain any useful information in order to avoid data privacy leakage.

The improvement of the literature [20-22] focuses mainly on the evidence generation algorithm, using the parameters γ, r to hide the evidence content, that is, the evidence content $\mu = \sum_{i=s_1}^{s_i} v_i$. The calculation of $v_i m_i$ is changed to $\mu' = \gamma \sum_{i=s_1}^{v_i} v_i + r$, cancel parameter μ is the linear combination in the calculation process. However, although this verification method can achieve privacy protection, the problem is that it cannot realize the fairness of the subsequent service payment of the verification scheme. If CSS and TPA collide to attack, that is, regardless of whether the cloud data is stored correctly, TPA will return complete information that the data have been saved. In the integrity verification process, DO has paid the corresponding fees to cloud service providers and verifiers in advance, so the fair payment content of the verification scheme is not guaranteed.

3. Verification scheme that supports privacy protection and fair payment

To solve the untrustworthy problem of TPA in actual integrity verification and realize service payment fairness, this section proposes a data integrity verification model that supports privacy

protection and fair payment. The verification model can be divided into three stages: initialization stage, audit stage, and punishment stage. As shown in Figure 2, the four entities included in the model are: DO, CSS, TPA, and SC. In the audit phase, the noninteractive verification mode NIDPDP is used to protect data privacy, and in the penalty phase, the fairness of service payment is realized through the penalty control of CSS and TPA by smart contracts.

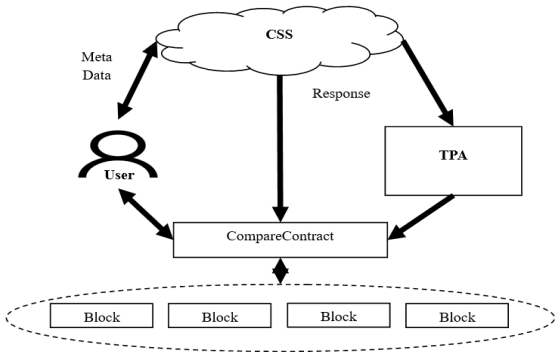

Figure 2: Integrity verification model supporting privacy protection and fair payment

3.1. Hierarchical Merkle Hash Tree

To ensure the integrity of data blocks in the cloud storage location and the verifiability of dynamic updates, we introduce a new type of data structure rank-based Merkle-Hash tree (RBMT). Compared to table-based skip movement, RBMT can not only fully support dynamic operations, including operations such as updating, deleting, and inserting data, but can also avoid the problem of excessive system communication overhead due to redundant authentication information. In RBMT, node W can be made up of three elements constitute, namely $W = \{r, s, h\}$. Among them, r is the level value of the node, indicating the number of leaf nodes that the current node can reach; s is the edge information of the storage node, indicating that the current node is the left/right child node of the parent node; the hash value h is the direct result of the leaf node mi. Hash, the hash value h of a nonleaf node is the result of concatenating the hash value of its left and right child nodes and its rank value and then hashing. The hash value of any node W is calculated as

$$h = \begin{cases} H(W), & W \text{ "is a leaf node",} \\ H(W.left.h \parallel W.right.h \parallel r) & \\ W \text{ "not a leaf node".} \end{cases} \quad (2)$$

When the verifier verifies whether the CSS correctly stores the location of the m3 data block, it uses the auxiliary authentication information Ω_3={W_4, W_c, Wb to calculate, as shown in Figure 3, the verification process has four steps:

(i). At node d, calculate the hash value of node d according to W_3 and $W_4 h_d = H(h_3 \| h_4 \| r_d)$;

(ii). At node a, calculate the hash value of node a according to Wc and Wd, $h_a = H(h_c \| h_d \| r_a)$

(iii). At root node root, calculate the hash value $h_{\text{root}} = H(h_a \| h_b \| r_{\text{root}} |)$ of the root node according to Wa and Wb, and obtain the root node W'root.

(iv). The verifier compares W'root with Wroot, if Wroot = W'root, the CSP stores the data correctly; otherwise, the CSS does not store the data correctly.

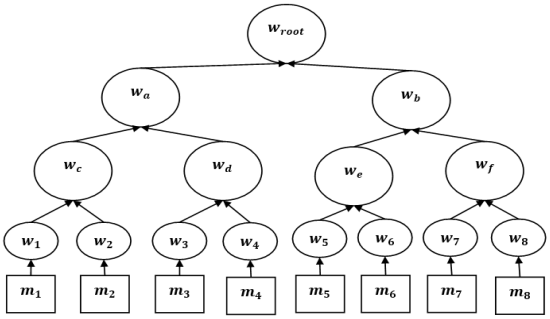

Figure 3: RBMT data location verification

When DO makes updates to data stored in the cloud, including Data block modification, data block deletion, and data

block insertion operations, the verification update process is:

(i). DO issues an update request In $f_o = (Up \text{ date }, i \cdot m_i^*, \sigma_i^*)$

(ii). CSS performs data update operations, and the update is divided into three stages. In the first stage, the content in the update request is parsed; in the second stage, the RBMT is updated according to the parsing result; in the third stage, the CSS returns the new root node $W_{r\,\text{root}}^*$ and auxiliary authentication information Ω_i to the verifier.

(iii). The verifier repeats the above RBMT data location verification process, and the completed data can be verified and updated dynamically.

3.2. No Interactive Dynamic Data Integrity Proof Mechanism

To protect user data privacy, this paper designs a non-interactive dynamic data integrity proof mechanism NIDPDP, as shown in Figure 4. In the audit phase of the scheme, the non-interactive verification mode is adopted, and the TPA does not need to send a challenge request to the CSS.

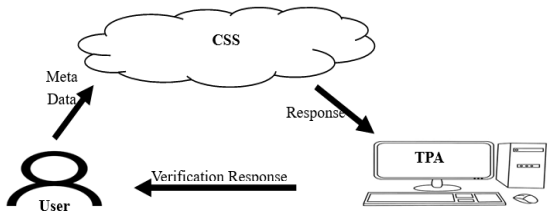

Figure 4: No Interactive Dynamic Data Possession Proof Mechanism

It is impossible to steal user data privacy by repeatedly challenging some data blocks in the same position during the verification process. In the audit stage, CSS and TPA respectively pass NIDPDP Proof Gen(). The NIDPDP Verify() process completes the task independently, reduces the information interaction between entities, and avoids the dishonest behavior of TPA in Section 1.2. At the same time, the NIDPDP mechanism also cooperates with the SC in the punishment phase, including the root node of the RBMT, the auxiliary authentication information, and the NIDPDP transmitted by the CSS to the contract. In the verification process (), TPA transmits the verification result to the contract.

In the audit phase, the operation process of the NIDPDP mechanism is as follows:

(i). CSS and TPA run the pseudo-random function $\varphi_Z(\tau)$ at the same time, and randomly select c elements from $[1, n]$ to form a subset $I = \{s_1, s_2, \ldots, s_c\}$. For each element $s_i \in I$, choose an integer $v_i = h(\tau \| i)$, which constitutes the challenge information $\text{Chal} = \{i, v_i\}_{s_1 \leqslant i \leqslant s_c}$. Among them, τ is the information that changes with time and is not controlled by CSS or TPA.

(ii). NIDPDP Evidence Generation. ProofGen(), CSS uses the public key pk, file F, authentication metadata set Φ and challenge information Chal as the input of the algorithm, and outputs the integrity proof P of the content of the verification data.

(iii). NIDPDP evidence verification. Verify(), TPA takes the public key pk and the integrity proof P as the input of the algorithm, and verifies whether the content proof P returned by CSS is correct. If the verification is successful, it returns result=1, if it fails, it returns result=0, and the verification result is sent to SC.

(iv). Execute the update NIDPDP. ExUpdate(), the algorithm is run by CSS to update the integrity proof, public key pk, data block set F, authentication metadata set Φ and update request Update as input, and return to the updated proof TPA PUpdate.

(v). Update and verify NIDPDP. VerUpdate(), the algorithm is run by TPA, the public key pk, the update

request Update, and the updated evidence P_{Update} are used as inputs to perform the update verification operation, and send the update verification result to SC.

3.3. Security Model

In a noninteractive data integrity verification scheme, the security model mainly consists of two objects, namely CSS and untrusted TPA.

1) Security verification of CSS

Define the data integrity proof game: CSS plays the role of adversary A and trusted entities play the role of challenger C. The game content is as follows:

Challenger C runs the key generation algorithm, and adversary A obtains the challenger's public-key information.

Attainment A and challenger C jointly run the pseudorandom function φZ (τ) to generate random challenge information Chal. The adversary A uses the challenge information Chal to initiate a data block label query to the challenger C, and challenger C calculates the corresponding set {σ1, σ2} adversary a.

The adversary A obtains the integrity evidence of the challenge data block set through calculation, and returns it to the challenger C, and the challenger verifies the evidence. If the verification is passed, adversary A wins.

Definition 1. If the three processes of the security verification of CSS are correct, the NIDPDP mechanism is said to be secure: that is, there is no polynomial-time adversary A that can win the data integrity proof game with a non-negligible probability.

2) Security verification of TPA

Define the data integrity proof game: The TPA plays the role of adversary A and trusted entities play the role of challenger C. The game content is as follows:

(a). A adversary and challenger C jointly run the pseudo-random function φZ (τ), and randomly select c elements from [1, n] to form a subset $I = \{s_1, s_2, \ldots., s_c\}$.;

(b). For each element $s_i \in I$, select an integer $v_i = h(\tau \| i)$, which constitutes the challenge information Chal = $\{i, v_i\}_{s_1 \leqslant i \leqslant s_c}$

(c). Adversary A runs the evidence generation algorithm NIDPDP. ProofGen(pk,F,Φ,Chal), get the challenge data block integrity proof P, and return this proof to challenger C;

(d). Challenger C runs the NIDPDP Verify evidence verification algorithm (pk, P) and outputs the verification result 0 or 1, of which 1 means the verification is passed;

(e). If adversary A is safe and reliable, then the output result of challenger C is 1.

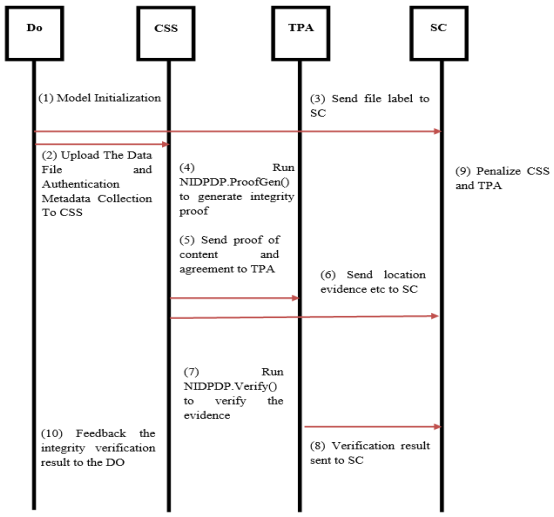

Figure 5: Integrity verification process supporting privacy protection and fair payment

Definition 2. If the five TPA security verification processes are correct, then the scheme model is said to be verifiably

secure to TPA. $\{m_1, m_2, \ldots, m_n\}$, meanwhile, each data block mi is divided into s segments, $m_i = \{m_{i1}, m_{i2}, \cdots, m_{is}\}$, therefore, the data file can be divided into $n \times s$ parts. Then, select the secret values $\tau_{1,}\tau_2, \ldots\ldots\ldots \tau_s \in Z_p$ of the preprocessed file, calculate the public verification parameter $u_j = g^{\tau_j} \in \mathbb{Z}_p$, and generate a corresponding label for each data block mi:

$$\sigma_i = H(\text{ Fname } \| i)^{\alpha} \cdot g_{j=1}^{\sum_{j=1}^{s} \tau_{ij} m_{ij}} \quad (3)$$

Among them, $\varepsilon = \sum_{j=1}^{s} \tau_j$. For file F, the file label t is the concatenation of the file identifier Fname and its signature, that is $t \leftarrow$ Fname $\|$ Sig_{ssk}(Fname), and ssk is the signature private key of the signature. Finally, the DO sends the file label to SC, uploads the F data file, and the set of authentication metadata $\Phi = \{\sigma_1, \sigma_2, \cdots, \sigma_n\}$. After that, delete the local file and the authentication metadata collection.

Audit stage: Both CSS and TPA run a pseudo-random function φZ (τ), randomly select c elements from [1,n] to form a subset I = s_1, s_2,, s_c for each element 〚 s〛_i∈I, select integers v_i=h(τ‖i), which constitute the challenge information, where τ is the information that changes with time and is not controlled by CSS or TPA.

1) 1) After the CSS generates the challenge information $Chal$, the parameters γ and λj are randomly selected, where $j \in [1, s], \pi \leftarrow e\left(\prod_{j=1}^{s} u_j^{\lambda_j}, H_2\right)$, which constitutes the protocol $C = \left(H_1^{\gamma}, \pi\right)$. Then, CSS runs NIDPDP according to the challenge information $Chal = \{i, \mathcal{V}_i\}_{s_1 \leqslant i \leqslant s_c}$. ProofGen() calculates the proof of integrity verification content:

$$\sigma = \prod_{i=s_1}^{s_i} \sigma_i^{\gamma_{v_i}}, \quad (4)$$

$$\mu_j = \lambda_j + \gamma \sum_{i=s_1}^{s_c} v_i m_{ij}, \mu = \{\mu_j\}_{j \in [1,s]}. \quad (5)$$

Finally, CSS constructs RBMT and sends the location evidence, i.e. the generated root node and the auxiliary authentication information corresponding to the challenge set, to SC.

2) TPA runs NIDPDP after receiving the content evidence sent by CSS. Verify(), that is, to verify whether formula (6) holds:

$$\pi \cdot e(\sigma, g) = e\left(\prod_{i=s_1}^{s_c} H(\text{ Fname } \| i)^{v_i}, H_1^{\gamma}\right) \cdot e\left(\prod_{j=1}^{s} u_j^{\mu_j}, H_2\right) \quad (6)$$

Send the verification result result=1 or result=0 to SC.

Punishment stage: The blockchain smart contract (shown in Figure 6(a)) will verify whether CSS and TPA have dishonest behaviours, in order to achieve fair payment of the verification scheme. If the CSS does not correctly store the DO data, CompareContract will call the contract TCSS (as shown in Figure 6(b)), and the CSS will pay the corresponding fine to the DO; similarly, if the TPA does not correctly verify the evidence returned by the CSS, the Compare Contract will call the contract TTPA (shown in Figure 6(c)), and TPA pays the corresponding fine to the DO. The specific punishment process is as follows:

1) In the audit stage, while sending the generated evidence to the TPA, CSS calls the smart contract Compare Contract that has been created on the blockchain and sends the constructed RBMT root node Wroot and auxiliary authentication information Ω_i (1⩽i⩽c). At the same

time, after calculating the verification result of the CSS evidence returned, TPA will also send the verification result to the blockchain smart contract CompareContract.

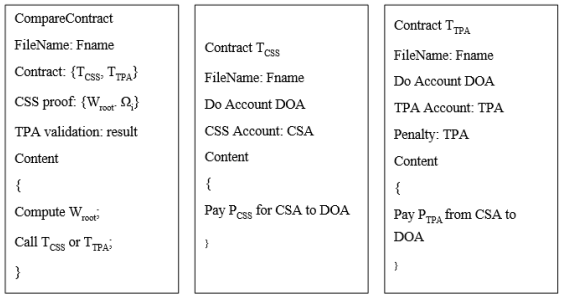

Figure 6: Smart Contract

2) The smart contract Compare Contract verifies the signature of the DO file first and terminates execution if the signature is invalid; otherwise, the challenge block index is also generated according to the pseudo-random function $\varphi Z\ (\tau)$. Compare Contract sends auxiliary authentication information $\Omega_i (1 \leqslant i \leqslant c)$ through CSS to calculate the root value W 'root' of RBMT, and compare it with the root value W_{root} sent by CSS. If the two are different, call the contract TCSS, and CSS pays the corresponding fine PCSS to the DO; if the two are the same, the result will be based on the verification result of the data content sent by the TPA. If result=0, the contract TCSS is also called, and CSS pays the corresponding fine PCSS to DO; if result=1, both the location evidence and the content evidence of the data block are passed to the verification and it can be judged that the CSS stores the data correctly. For TPA, if the smart contract CompareContract verifies that the root value W_{root} returned by CSS is not equal to the calculated root value W'_{root}, TPA still returns the verification success, it can be inferred that TPA does not honestly execute the verification algorithm NIDPDP. Verify(), therefore, the smart contract compare contract will call the contract TTPA, and TPA will pay the corresponding penalty PTPA to DO

4. Performance analysis

4.1. Correctness analysis

Theorem 1. If CSS honestly stores user data and returns the corresponding evidence, then the content evidence can pass the verification of TPA, so Equation (6) is established.

Proof: Theorem 1 can be proved as follows:

$$\pi \cdot e(\sigma, g) = e\left(\prod_{j=1}^{s} u_j^{\lambda_j}, H_2\right) \cdot e\left(\prod_{i=s_1}^{s_c} \sigma_i^{\gamma_i}, g\right)$$

$$= e\left(\prod_{j=1}^{s} u_j^{\lambda_j}, H_2\right) \cdot e\left(\prod_{i=s_1}^{s_c} \left(H(\text{ Fname } \| i)^{\alpha} \cdot g_{j=1}^{s} \tau_j m_{ijj\beta}^{\beta}\right)^{\gamma_{v_i}}, g\right)$$

$$= e\left(\prod_{j=1}^{s} u_j^{\beta\lambda_j}, g\right) \cdot e\left(\prod_{i=s_1}^{s_c} H(\text{ Fname } \| i)^{\alpha\gamma_{v_i}} \prod_{j=1}^{s} \sum_{\substack{i=s_1 \\ j}}^{s_{ij}\beta\gamma_{v_i}}, g\right)$$

$$= e\left(\prod_{i=s_1}^{s_c} H(\text{ Fname } \| i)^{\alpha\gamma_{v_i}} \prod_{j=1}^{s} u_j^{\beta(\lambda_{j+\gamma} \sum_{i=s_1}^{s_c} m_{ij} v_i)}, g\right)$$

$$= e\left(\prod_{i=s_1}^{s_c} H(\text{ Fname } \| i)^{\alpha\gamma_{v_i}} \prod_{j=1}^{s} u_j^{\beta\mu_j}, g\right)$$

$$= e\left(\prod_{i=s_1}^{s_c} H(\text{ Fname } \| i)^{v_i}, g^{\alpha\gamma}\right) \cdot e\left(\prod_{j=1}^{s} u_j^{\mu_j}, g^{\beta}\right)$$

$$= e\left(\prod_{i=s_1}^{s_c} H(\text{ Fname } \| i)^{v_i}, H_1^{\gamma}\right) \cdot e\left(\prod_{j=1}^{s} u_j^{\mu_j}, H_2\right).$$

4.2 Security analysis

Theorem 2. If the label generation algorithm of this scheme is unforgeable, and the CDH-hard problem and the DL-hard problem are unsolvable under the random oracle model, then there is no polynomial-time algorithm that can destroy the security of the model with a non-negligible probability.

Proof: To prove the correctness of the theorem, the following games have been considered.

Game1: Defined as a verifiable secure data integrity proof game for CSS.

The adversary A obtains the label set corresponding to the file data block through the query and generates the integrity evidence corresponding to the challenge set. Challenger C uses the evidence verification algorithm to verify the evidence, if the verification is successful, then adversary A wins the game.

Game2: Game2 is similar to Game1, the difference being that Challenger C will keep all the data block tag proofs that have been interrogated by the adversary. If the evidence generated by adversary A can pass the verification, but the label evidence σ' contained in the evidence is not equal to the value of σ returned by the honest prover, then challenger C will terminate the game. Assuming that Chal $= \{i, v_i\}_{s_1 \leqslant i \leqslant s_c}$ is the challenge set leading to the termination of the game, the evidence returned by the honest prover is $\theta = \{\sigma, \mu\}$, and the evidence returned by the adversary A is $\theta' = \{\sigma', \mu'\}$. According to the definition of Game2, the evidence

returned by the adversary can be verified by equation (7):

$$\pi \cdot e(\sigma, g) = \&e\left(\prod_{i=s_1}^{s_c} H(\text{Fname} \parallel i)^{v_i}, H_1^{\gamma}\right) \cdot e\left(\prod_{j=1}^{s} u_j^{\mu_j}, H_2\right) \quad (7)$$

Since the game has been terminated by challenger C, it can be known that σ′≠σ, and θ′ can pass the verification. Therefore, $\mu' \neq \mu$, that is, the condition of $\Delta\mu = \mu' - \mu \neq 0$ is established.

In Game2, the trusted entity plays the challenger C, and the interaction process with the adversary A is as follows:

(a). Challenger C runs the key generation algorithm KeyGen(1k), and adversary A obtains the challenger's public key information.

(b). The adversary A initiates a data block label query to challenger C and obtains the set {σ_1, σ_2, list of labels.

(c). The adversary A calculates the content evidence of the challenge data block through the evidence generation algorithm and returns it to challenger C, and the challenger verifies the evidence.

(d). Process (b) and Process (c) are repeated until the evidence returned by adversary A is verified, but the evidence contains label evidence σ' $\prod_{j=1}^{s} u_j$that does not belong to the label list saved by challenger C. Given, $u_j = g^{\tau_j}$. Here Challenger C's goal is to get $\prod_{j=1}^{s} g^{\tau_j \beta}$ Challenger C by calculating:

$$e\left(\frac{\sigma'}{\sigma}, g\right) = e\left(\prod_{j=1}^{s} u_j^{\beta\Delta\mu_j}, g\right) = e\left(\prod_{j=1}^{s} g^{\tau_j \beta \Delta\mu_j}, g\right), \quad (8)$$

$$\prod_{j=1}^{s} g^{\tau_j \beta} = \frac{\sigma'}{\sigma^{j=1} \prod_{j=\mu}^{s}}. \quad (9)$$

To analyze the probability that adversary A wins Game 2, only points are needed. The probability that $\prod_{j=1}^{s} \Delta\mu_j = 0$modp is established, and the probability of its success is $1 / p$, which can be ignored.

The interaction process (a) and the process (b) of the trusted entity acting as challenger C and adversary A in Game3 are the same as in Game2. For process (c), given $\prod_{j=1}^{s} u_j, g$, the challenger C's goal is to obtain $\tau_1, \tau_2, \cdots, \tau_s$ such that $\prod_{j=1}^{s} u_j = g_{j=1}^{\sum_{j=1}^{s} \tau_j}$. Challenger C by calculating:

$$\begin{cases} \pi \cdot e(\sigma, g) = e\left(\prod_{i=s_1}^{c_c} H(\text{Fname} \parallel i)^{v_i}, H_1^{\gamma}\right) \cdot e\left(\prod_{j=1}^{s} u_j^{\mu_j}, H_2\right) \\ \pi \cdot e(\sigma', g) = e\left(\prod_{i=s_1} H(\text{Fname} \parallel i)^{v_i}, H_1^{\gamma}\right) \cdot e\left(\prod_{j=1}^{s} u_j^{\mu_j'}, H_2\right) \end{cases} \quad (10)$$

$$g_{j=1}^{\sum_j^s \tau_j \Delta\mu_j} = 1 \Rightarrow g_{j=1}^{\sum_{j=1}^{s} \tau_j \Delta\mu_j} = 1 \quad (11)$$

$$\sum_{j=1}^{s} \tau_j = g^{\frac{1}{\Delta\mu_j}} \quad (12)$$

To analyze the probability that adversary A wins Game 3, we only need to analyses that the probability of $\Delta\mu j = 0\ mod\ p$ is 1⁄p, which can be ignored. Therefore, if there is an adversary A that can win Game2 and Game3 with negligible probability, then there will be a trusted entity to solve the hard problem of DL.

To sum up, if the label generation algorithm of this scheme is unforgeable, then there is no polynomial-time algorithm that can win Games 1, 2, and 3 with a nonnegligible probability, destroying the security of the model.

4.3 Privacy analysis

Theorem 3. *In the data integrity verification scheme that supports privacy protection and fair payment proposed in this document, given the content evidence information θ={σ, μ} returned by CSS, if a*

curious and dishonest TPA tries to obtain DO from the information it has already grasped, the data information $F = \{m_1, m_{2,}, \ldots\ldots, m_n\}$ *is computationally infeasible.*

Proof: As can be seen in Figure 5, in the data integrity verification in the audit stage, TPA accepts the evidence information returned by CSS and passes the verification algorithm NIDPDP. Verify(pk,θ) judges whether the returned evidence is correct. However, even if CSS returns evidence $\sigma = \prod_{i=s_1}^{s_c} \sigma_i^{\gamma_i}$ and $\mu = \sum_{i=s_1}^{s_c} v_i m_i$ is about the linear combination of DO original data. The curious TPA makes use of this feature and tries to repeatedly detect data blocks in some positions through complex calculation to form a linear equation system and obtain DO original data by solving the equation system. However, due to the limitation of the NIDPDP mechanism, Chal challenge information is generated randomly and is not controlled by TPA. CSS and TPA pass through NIDPDP, respectively. ProofGen(), NIDPDP. The Verify() process completes the task independently, so it is impossible for a curious TPA to crack user data through replay attacks. Also, in the penalty phase the smart contract Compare Contrac is under the constraints of, because TPA cannot judge the correctness of the location evidence returned by CSS to Compare Contract, once the smart contract Compare Contract judges that the location evidence returned by CSS to it is wrong, and TPA returns the correct verification result, it will call. The contract TTPA punishes the TPA. Therefore, TPA and CSS will perform their tasks independently in integrity verification, and it is impossible to collaborate to attack. To sum up, in the integrity verification scheme proposed in this paper, the curious and dishonest TPA cannot obtain the user's data information, which effectively guarantees the user's privacy security.

5. Experiments and Results

In this section, the performance of the proposed scheme will be evaluated from the aspects of computational overhead and communication overhead.

5.1. Computational overhead

The computational overhead comes mainly from the three entities of the scheme, including DO, TPA, CSS, and the computational overheads they generate at different stages determine the verification efficiency of the entire scheme. In the initialization phase, it is mainly the overhead generated by DO to generate tags for file data blocks. In the audit stage, the computational overhead is mainly generated by CSS generation evidence and TPA verification evidence. To evaluate the performance of the scheme model, we first analyze the impact of different data block sizes on the computational cost of the proposed scheme. The experimental environment is configured as Windows 10 system, Intel Core i5 processor, 2.20GHz CPU, 4GB RAM. The scheme uses Java language to perform basic functions, and the encryption and decryption algorithm is called from the JPBC library, and an average of 10 experimental results are taken. In the audit stage, the sampling strategy is adopted and the number of challenge blocks is selected from the total number of file data blocks with a probability of 4.6%. The file data block sizes are 30KB, 60KB, 90KB, 120KB, and 150KB, 180KB, 210KB, 240KB, 270KB, 300KB respectively. In the case of 64MB, 128MB, 256MB, 512MB, 1024MB, random files are used to test the impact of

data block size on the computational cost of each entity in different verification stages.

It can be seen from Figure 7 and Table 1 that with increasing data block, the time it takes for DO to generate the authentication metadata tag gradually decreases, and after 240KB, the tag generation time is basically stable at a certain value. This is because, with increasing the data block, the number of file data blocks decreases and the number of tags also decreases, but when the size of the data block reaches a certain value, the number of generated tags is not much different. The number of data segments also increases, and the calculation time of a single tag becomes longer. Therefore, when the size of the data block increases to some extent, the tag generation time will stabilize at a certain value. As shown in Figure 8 and Table 2, with increasing data block, the CSS generated time to complete the evidence also gradually increases. This is because as the data block increases, the number of segments in the data block s increases, and the number of elements μj contained in the integrity evidence generated by CSS also increases. Therefore, the more complicated the evidence calculation, the more time it takes. As shown in Figure 9 and Table 3, with increasing data block, the time for TPA to verify evidence gradually decreases. This is because the number of challenge blocks c decreases with the increase of data blocks, and the computational cost of TPA verification integrity, proof is $(c+2)M+(s+c+1)E+2P$), so the TPA verification time will decrease.

Table 1: Data Block Size and the Time

Data Size	1024KB	512KB	256KB	128KB	64KB
30	35	28	12	6	2
60	20	23	11	5.5	1.9
90	18	17	10	5	1.7
120	16	15	9.5	4.3	1.6
150	13	11	9	4	1.5
180	9	8	7	3.5	1.4
210	7	5	4	3	1.3
240	5	4	2	2.4	1.2
270	4	2	1	2	1.1
300	2	1.8	1.5	1.2	1

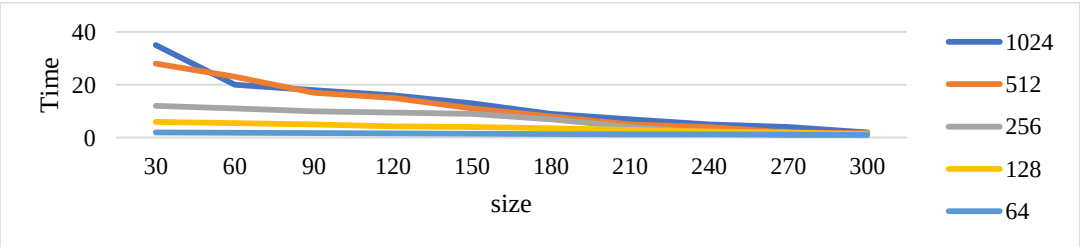

Figure 7: The Influence of Data Block Size on the Time of DO Label Generation

In summary, in order to optimize the performance of the proposed scheme, we choose the ideal case where the block is 240KB, and the system overhead of the

verification process of data files of different sizes is calculated.

Table 2: CSS generation data

Data Size	1024 KB	512 KB	256 KB	128 KB	64 KB
30	2.8	2.5	1.4	1.2	1
60	3	2.6	1.6	1.3	1.2
90	3.5	2.8	1.8	1.5	1.3
120	3.9	2.9	1.9	1.6	1.4
150	4.2	3.2	2.2	1.6	1.6
180	4.5	3.7	2.3	1.7	1.7
210	4.9	3.9	2.4	1.8	1.8
240	5.8	4	2.6	1.9	1.9
270	7.2	4.2	2.8	2.1	1.9
300	7.9	4.6	3	2.1	1.9

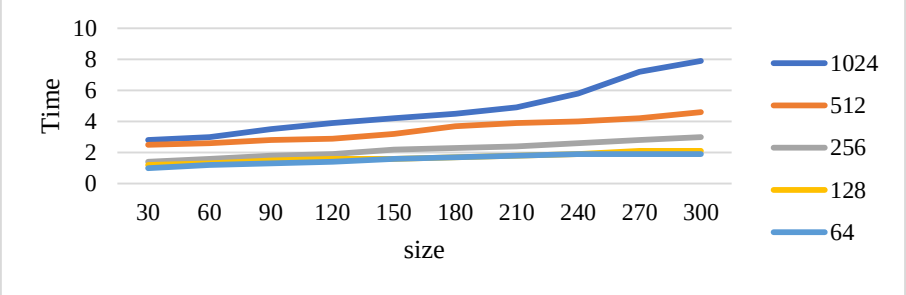

Figure 8: The Effect of Data Block Size on CSS Generation Time

Reference [32] adopts the TPA-based verification mechanism, and the scheme mainly includes two stages: initialization stage and audit stage. To prove the superiority of the performance of the proposed scheme, we compare the computational overhead and communication overhead with the literature [32]. At the same time, reference [32] supports global and sampling verification, and sampling verification also challenges data blocks with a fixed probability of 4.6%. The comparison of theoretical calculation cost between the proposed scheme and the literature [32] is shown in Table 4, where n represents the total number of data blocks in the file, c represents the number of challenge blocks, s represents the number of segments contained in the data block, M represents the multiplication operation, E represents the exponential operation and P represents the bilinear map.

It can be seen that although the overhead of label generation and the overhead of evidence generation of the literature [32] are smaller than those of this scheme. Since the time taken by the bilinear operation is much longer than the multiplication operation and the exponential operation, and the TPA verification evidence is performed periodically, the bilinear scheme proposed in this paper is much shorter than the literature [32], and the computational cost is lower. During the experiment, 64MB, 128MB, 256MB, 512MB, and a random file of 1024MB are used to compare the computational cost of the document [32] and the scheme in this paper. Among them, the number of challenge blocks represents 4.6% of the total number of file

data blocks. The experimental results are shown in Figure 10 and Table 5.

Table 3: TPA verification proof data

Data Size	1024 KB	512 KB	256 KB	128 KB	64 KB
30	2.5	2.3	1.2	0.6	0.2
60	1.9	1.7	1.1	0.55	0.19
90	1.8	1.6	1	0.5	0.17
120	1.5	1.1	0.95	0.43	0.16
150	1.2	1	0.9	0.4	0.15
180	0.9	0.5	0.7	0.35	0.14
210	0.7	0.4	0.4	0.3	0.13
240	0.5	0.3	0.2	0.24	0.12
270	0.4	0.2	0.1	0.2	0.11
300	0.2	0.16	0.15	0.12	0.1

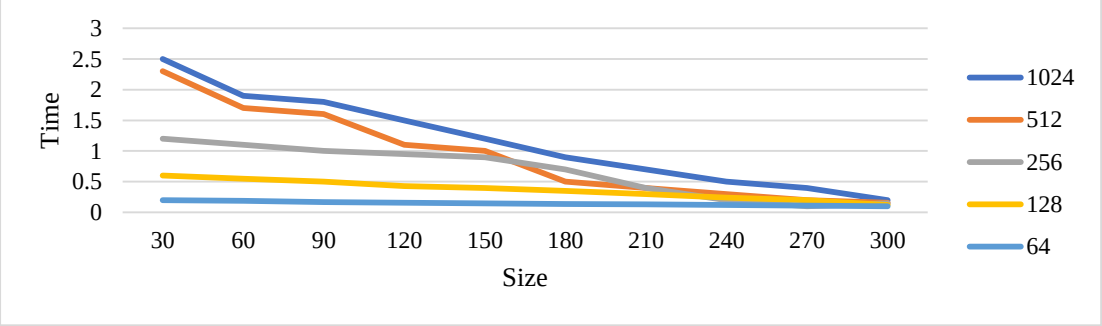

Figure 9: The Effect of Data Block Size on TPA Verification Proof Time

Table 4: Computational Cost Comparison

Index	Existing method	Proposed Method
DO generation mark	$nM + 2nE$	$n(s+1)M + 2nE$
CSP generates evidence	$(2c-1)M + cE$	$(s \times c + c - 1)M + cE$
TPA verification evidence	$cM + E + (c+1)P$	$(c+2)M + (s+c+1)E + 2P$

Table 5: Computation Cost

Data Size	Existing Method	Proposed Method
64	1.1	0.8
128	2.3	1.8
256	3.2	2.7
512	6.5	5.8
1024	9.2	8.7

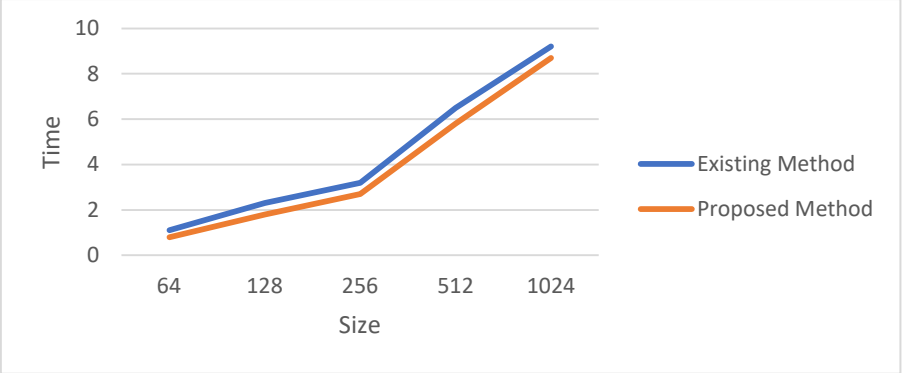

Figure 10: Comparison of the computational cost of the two schemes

5.2. Communication Overhead

The overhead of the communication process mainly lies in the information exchange, and in the cloud storage integrity verification model, it mainly depends on the information exchange among CSS, TPA, and SC in the audit stage. Specifically, the communication overhead in this scheme includes the process of CSS sending content integrity evidence to TPA and sending location integrity evidence to SC in the audit stage, and the process of TPA sending content verification result to SC. As shown in Table 6, the communication costs are 0, O(log n/c), O(1). Compared to the literature [32], this scheme does not have the TPA challenge request process sent to CSS throughout the verification process, so the communication overhead of the scheme will be greatly reduced. Figure 11 and Table 7 show the results of the communication overhead experiment comparison between the literature [32] and the proposed scheme.

Table 6: Comparison of Communication Overhead

Index	Existing method	Proposed Method
TPA sends challenge information	$O(c)$	0
CSS returns complete certificate	$O(1)$	$O(\log\frac{n}{c})$
TPA returns content verification results	0	$O(1)$

Table 7: Communication Overhead

Data Size	Existing Method	Proposed Method
64	7	0.8
128	11	1.8
256	18	3.4
512	28	4.3
1024	54	5.6

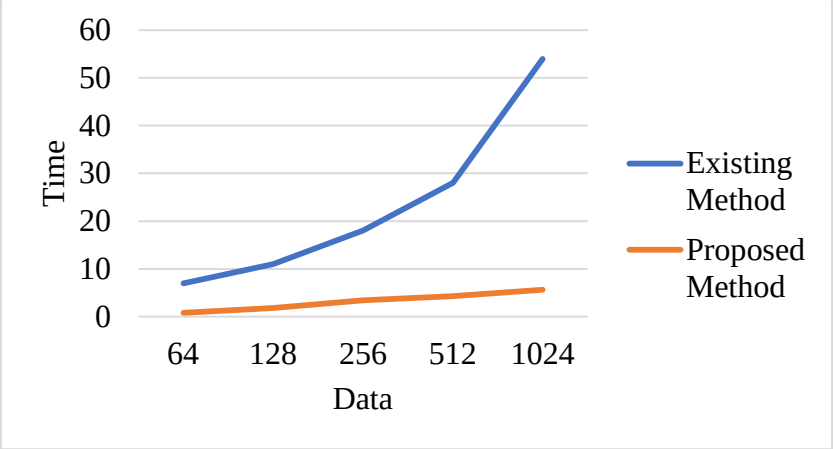

Figure 11: Comparison of the Communication Overhead of the Two Schemes

6. Conclusions

This paper proposes a data integrity verification scheme that supports privacy protection and fair payment, which can solve the problems of privacy leakage and fair payment in the data integrity verification process. In order to protect user data privacy, this paper designs a non-interactive dynamic data integrity proof mechanism NIDPDP. In the audit stage, TPA does not need to send a challenge request to CSS, that is, the challenge interaction process between TPA and CSS is cancelled to avoid user data privacy of leakage. In order to achieve fair service payment, SC first calculates the RBMT root node through auxiliary authentication information and compares it with the root node sent by CSS to ensure the integrity of the data block in position. Second, the blockchain smart contract Compare Contract punishes the dishonest behavior of CSS and TPA by calling the contracts TCSS and TTPA, so that each entity can honestly execute according to the rules of the agreement. Theoretical analysis and experimental results show that, compared with other schemes, the proposed scheme can further reduce computational overhead and communication overhead and can realize the fairness of service payment while ensuring data privacy.

References

[1]. J. Shen, T. Zhou, D. He, Y. Zhang, X. Sun and Y. Xiang, "Block Design-Based Key Agreement for Group Data Sharing in Cloud Computing," in IEEE Transactions on Dependable and Secure Computing, vol. 16, no. 6, pp. 996-1010, 1 Nov.-Dec. 2019, DOI: 10.1109/TDSC.2017.2725953.

[2]. M. Bahrami, "Cloud Computing for Emerging Mobile Cloud Apps," 2015 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, 2015, pp. 4-5, DOI: 10.1109/MobileCloud.2015.40.

[3]. P. Dutta, T. Mukherjee, V. G. Hegde and S. Gujar, "C-Cloud: A Cost-Efficient Reliable Cloud of Surplus Computing Resources," 2014 IEEE 7th International Conference on Cloud Computing, 2014, pp. 986-987, DOI: 10.1109/CLOUD.2014.152.

[4]. Parashar, A., Parashar, A., Ding, W., Shekhawat, R. S., & Rida, I. (2023). Deep learning pipelines for recognition of gait biometrics with

covariates: A comprehensive review. Artificial Intelligence Review, 1-65.

[5]. P. Sharma and V. Jadhao, "Molecular Dynamics Simulations on Cloud Computing and Machine Learning Platforms," 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), 2021, pp. 751-753, DOI: 10.1109/CLOUD53861.2021.00101.

[6]. V. Marbukh, "Systemic Risks in the Cloud Computing Model: Complex Systems Perspective," 2016 IEEE 9th International Conference on Cloud Computing (CLOUD), 2016, pp. 863-866, DOI: 10.1109/CLOUD.2016.0124.

[7]. D. S. Linthicum, "Connecting Fog and Cloud Computing," in IEEE Cloud Computing, vol. 4, no. 2, pp. 18-20, March-April 2017, DOI: 10.1109/MCC.2017.37.

[8]. R. Rauscher and R. Acharya, "Virtual Machine Placement in Predictable Computing Clouds," 2014 IEEE 7th International Conference on Cloud Computing, 2014, pp. 975-976, DOI: 10.1109/CLOUD.2014.148.

[9]. S. Tuli, S. Ilager, K. Ramamohanarao and R. Buyya, "Dynamic Scheduling for Stochastic Edge-Cloud Computing Environments Using A3C Learning and Residual Recurrent Neural Networks," in IEEE Transactions on Mobile Computing, vol. 21, no. 3, pp. 940-954, 1 March 2022, DOI: 10.1109/TMC.2020.3017079.

[10]. C. Lin and S. Lu, "Scheduling Scientific Workflows Elastically for Cloud Computing," 2011 IEEE 4th International Conference on Cloud Computing, 2011, pp. 746-747, DOI: 10.1109/CLOUD.2011.110.

[11]. Li, W., Ni, L., Wang, J., & Wang, C. (2022), Collaborative representation learning for nodes and relations via heterogeneous graph neural network, Knowledge-Based Systems, 255, 109673, 2022.

[12]. Gera, T., Singh, J., Mehbodniya, A., Webber, J. L., Shabaz, M., & Thakur, D. (2021). Dominant feature selection and machine learning-based hybrid approach to analyze android ransomware. Security and Communication Networks, 2021, 1-22.

[13]. H. Kim, J. Shin, Y. Song and J. Chang, "Privacy-Preserving Association Rule Mining Algorithm for Encrypted Data in Cloud Computing," 2019 IEEE 12th International Conference on Cloud Computing (CLOUD), 2019, pp. 487-489, DOI: 10.1109/CLOUD.2019.00086.

[14]. Y. Yang, "Towards Multi-user Private Keyword Search for Cloud Computing," 2011 IEEE 4th International Conference on Cloud Computing, 2011, pp. 758-759, DOI: 10.1109/CLOUD.2011.76.

[15]. Y. Amanatullah, C. Lim, H. P. Ipung and A. Juliandri, "Toward cloud computing reference architecture: Cloud service management perspective," International Conference on ICT for Smart Society, 2013, pp. 1-4, DOI: 10.1109/ICTSS.2013.6588059.

[16]. Mehbodniya, A., Webber, J. L., Neware, R., Arslan, F., Pamba, R. V., & Shabaz, M. (2022), Modified Lamport Merkle Digital Signature blockchain framework for

authentication of internet of things healthcare data, Expert Systems, 39(10), e12978.

[17]. M. Bahrami and M. Singhal, "A dynamic cloud computing platform for eHealth systems," 2015 17th International Conference on E-health Networking, Application & Services (HealthCom), 2015, pp. 435-438, DOI: 10.1109/HealthCom.2015.7454539.

[18]. B. Snyder, R. Green, V. Devabhaktuni and M. Alam, "Evaluation of Highly Reliable Cloud Computing Systems Using Non-sequential Monte Carlo Simulation," 2014 IEEE 7th International Conference on Cloud Computing, 2014, pp. 940-941, DOI: 10.1109/CLOUD.2014.133.

[19]. P. Maenhaut, H. Moens, B. Volckaert, V. Ongenae and F. De Turck, "Resource Allocation in the Cloud: From Simulation to Experimental Validation," 2017 IEEE 10th International Conference on Cloud Computing (CLOUD), 2017, pp. 701-704, DOI: 10.1109/CLOUD.2017.96.

[20]. J. Y. Zhang, P. Wu, J. Zhu, H. Hu and F. Bonomi, "Privacy-Preserved Mobile Sensing through Hybrid Cloud Trust Framework," 2013 IEEE Sixth International Conference on Cloud Computing, 2013, pp. 952-953, DOI: 10.1109/CLOUD.2013.108.

[21]. Rida, I., Almaadeed, S., & Bouridane, A. (2014, December). Improved gait recognition based on gait energy images. In 2014 26th International Conference on Microelectronics (ICM) (pp. 40-43). IEEE.

[22]. D. Berzano, J. Blomer, P. Buncic, G. Ganis, G. Lestaris and R. Meusel, "Interactive Exploitation of Nonuniform Cloud Resources for LHC Computing at CERN," 2013 IEEE Sixth International Conference on Cloud Computing, 2013, pp. 976-977, DOI: 10.1109/CLOUD.2013.88.

[23]. M. Kretzschmar, M. Golling and S. Hanigk, "Security Management Areas in the Inter-cloud," 2011 IEEE 4th International Conference on Cloud Computing, 2011, pp. 762-763, DOI: 10.1109/CLOUD.2011.83.

[24]. J. Zhao, M. A. Rodríguez and R. Buyya, "A Deep Reinforcement Learning Approach to Resource Management in Hybrid Clouds Harnessing Renewable Energy and Task Scheduling," 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), 2021, pp. 240-249, DOI: 10.1109/CLOUD53861.2021.00037.

[25]. Y. Wadia, R. Gaonkar and J. Namjoshi, "Portable Autoscaler for Managing Multi-cloud Elasticity," 2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies, 2013, pp. 48-51, DOI: 10.1109/CUBE.2013.19.

[26]. I. Ayadi, N. Simoni and T. Aubonnet, "SLA Approach for "Cloud as a Service"," 2013 IEEE Sixth International Conference on Cloud Computing, 2013, pp. 966-967, DOI: 10.1109/CLOUD.2013.127.

[27]. D. S. Linthicum, "Emerging Hybrid Cloud Patterns," in IEEE Cloud Computing, vol. 3, no. 1, pp. 88-91,

Jan.-Feb. 2016, DOI: 10.1109/MCC.2016.22.

[28]. F. Yang-Turner et al., "Scalable Pathogen Pipeline Platform (SP^3): Enabling Unified Genomic Data Analysis with Elastic Cloud Computing," 2019 IEEE 12th International Conference on Cloud Computing (CLOUD), 2019, pp. 478-480, DOI: 10.1109/CLOUD.2019.00083.

[29]. D. S. Linthicum, "The Technical Case for Mixing Cloud Computing and Manufacturing," in IEEE Cloud Computing, vol. 3, no. 4, pp. 12-15, July-Aug. 2016, DOI: 10.1109/MCC.2016.75.

[30]. Y. Kaneko and T. Ito, A Reliable Cloud-Based Feedback Control System, 2016 IEEE 9th International Conference on Cloud Computing (CLOUD), 2016, pp. 880-883, DOI: 10.1109/CLOUD.2016.0128.

[31]. D. S. Linthicum, Cloud Computing Changes Data Integration Forever: What's Needed Right Now, in IEEE Cloud Computing, vol. 4, no. 3, pp. 50-53, 2017, DOI: 10.1109/MCC.2017.47.

[32]. J. Shen, J. Shen, X. Chen, X. Huang and W. Susilo, "An Efficient Public Auditing Protocol With Novel Dynamic Structure for Cloud Data," in IEEE Transactions on Information Forensics and Security, vol. 12, no. 10, pp. 2402-2415, Oct. 2017, DOI: 10.1109/TIFS.2017.2705620.

[33]. P. Geetha and C. R. R. Robin, "A comparative-study of load-cloud balancing algorithms in cloud environments," 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), 2017, pp. 806-810, DOI: 10.1109/ICECDS.2017.8389549.

[34]. Rida, I. (2018). Feature extraction for temporal signal recognition: An overview. arXiv preprint arXiv:1812.01780.

[35]. M. Joshi, K. Joshi and T. Finin, "Attribute Based Encryption for Secure Access to Cloud Based EHR Systems," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), 2018, pp. 932-935, DOI: 10.1109/CLOUD.2018.00139.

[36]. Y. Zhang, X. Li and H. Qian, "An anonymous remote attestation for trusted cloud computing," 2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, 2012, pp. 426-429, DOI: 10.1109/CCIS.2012.6664441.

[37]. J. Yang, L. Zhang and X. A. Wang, "On Cloud Computing Middleware Architecture," 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2015, pp. 832-835, DOI: 10.1109/3PGCIC.2015.46.

[38]. R. Jindal, N. Kumar and H. Nirwan, "MTFCT: A task offloading approach for fog computing and cloud computing," 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2020, pp. 145-149, DOI: 10.1109/Confluence47617.2020.9058209.

[39]. Li, W., Zhou, X., Yang, C., Fan, Y., Wang, Z., & Liu, Y. (2022). Multi-objective optimization algorithm based on characteristics fusion of

dynamic social networks for community discovery. Information Fusion, 79, 110-123.

[40]. C. Wu, A. N. Toosi, R. Buyya and K. Ramamohanarao, "Hedonic Pricing of Cloud Computing Services," in IEEE Transactions on Cloud Computing, vol. 9, no. 1, pp. 182-196, 1 Jan.-March 2021, DOI: 10.1109/TCC.2018.2858266.

[41]. D. Che, J. Fairfield, P. Ghodous and J. P. Gelas, "Enhanced Backfill Computing," 2014 IEEE 7th International Conference on Cloud Computing, 2014, pp. 956-957, DOI: 10.1109/CLOUD.2014.140.