IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received March 29, 2022, accepted May 15, 2022, date of publication May 20, 2022, date of current version May 31, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3176754

Data Integrity Audit Scheme Based on Blockchain Expansion Technology

ZHENPENG LIU[1], YONGJIANG FENG[1], LELE REN[1], AND WEIHUA ZHENG[2,3]
[1]School of Cyberspace Security and Computer, Hebei University, Baoding 071002, China
[2]School of Information Engineering, Handan University, Handan 056005, China
[3]School of Information Science and Engineering, Xinjiang University of Science and Technology, Korla 841000, China

Corresponding author: Weihua Zheng (gaoyang@hdc.edu.cn)

This work was supported in part by the National Natural Science Foundation of Hebei Province under Grant F2019201427, and in part by the Ministry of Education of China under Project 2017A20004.

ABSTRACT Increasing numbers of users are outsourcing data to the cloud, but data integrity is an important issue. Due to the decentralization and immutability of blockchain, more and more researchers tend to use blockchain to replace third-party auditors. This paper proposes a data integrity system based on blockchain expansion technology that aims to solve the problem of high cost for blockchain network maintenance and for user creation of new blocks caused by the rapid growth of blocks in the data integrity audit scheme of existing blockchain technology. Users and cloud service providers (CSP) deploy smart contracts on the main chain and sub-chains. Intensive and frequent computing work is transferred to the sub-chain for completion, and the computation results of the sub-chain are submitted to the main chain periodically or when needed to ensure its finality. The concept of non-interactive audit is introduced to avoid affecting user experience due to the communication with the CSP during the audit process. In order to ensure data security, a reward pool mechanism is introduced. Comprehensive analysis from aspects such as storage, batch auditing and data consistency proves the correctness of the scheme. Experiments on the Ethereum blockchain platform demonstrate that this scheme can effectively reduce storage and computational overhead.

INDEX TERMS Blockchain, cloud storage, data auditing, blockchain expansion.

I. INTRODUCTION

Cloud computing is a distributed computing model based on a large shared virtualized computing resource pool, it helps users use powerful computing and storage resources. And it can greatly reduce the burden of data storage on hardware and software for users, which encourages many enterprises and individuals to store their data on cloud servers [1].

Despite the great success of cloud storage, it also faces various challenges [2]–[4], and its security, reliability and privacy have always been a serious issue [6], [7]. After the user stores the data on the cloud server, the server provider may damage or delete the user data due to various factors [12], verifying the integrity of outsourced data becomes a crucial issue in cloud storage. Remote data integrity audit technology is very convenient and safe to help users check the integrity of data stored in outsourced [5], [28]. Therefore, the essence of cloud data security is how cloud storage providers (CSP) can establish trust with users. Cloud device failures, illegal attacks, and CSPs may be bribed to view user data, all of which can lead to illegal infringement of user data. Furthermore, even if the user data is damaged, the user may not be able to hold the CSP accountable effectively, since the CSP may evade responsibility and deny it [16]. This is due to the lack of trust between the two parties, resulting in the party being questioned being unable to come up with evidence that would convince the other party. In addition, the current law on cybersecurity is not sound, which makes it difficult for users to obtain due compensation [18].

In traditional cloud auditing schemes, there is an entity called auditors (often referred to as third-party auditors, or TPA) which implement public audits [8], [21]. The TPA accept audit mandates from data owners and perform as instructed. In each of these methods, a trusted Third Party Auditor (TPA) must be found to assist the user in auditing, but in reality it is difficult to find fully trusted third-party auditors. For example, TPA will also partner with CSP for some ulterior purpose to hide data corruption, or with data owners to avoid penalties.

The emergence of blockchain can solve this problem very well. Blockchain has the properties of decentralization, tamper resistance, consistency and traceability. Therefore,

The associate editor coordinating the review of this manuscript and approving it for publication was Thanh Ngoc Dinh.

information stored on the blockchain is open and transparent. In recent years, more and more researchers use blockchain to replace third-party auditors [9], [10]. Although the use of blockchain as a trusted third-party auditor can well address users' concerns in cloud computing environments, but the rapid growth of blocks will lead to high cost for blockchain network maintenance and for user creation of new blocks [17].

To solve the above problems, a data integrity verification scheme based on block chain expansion technology is proposed. By slowing the growth of the block chain, reducing storage and calculation costs. In particular, our contribution can be summarized in three aspects:

1) A data integrity audit protocol based on plasma smart contracts is proposed. By introducing plasma sub-chains and deploying smart contracts on the main chain and sub-chains, the storage pressure of the main chain can be reduced and the growth rate can be slowed down through this protocol. TPA audit protocol can be executed with low computational and communication overhead.
2) A batch auditing scheme is proposed, the scheme can batch-process multiple audit tasks at the same time. In order to avoid affecting the user experience due to the communication with the CSP during the audit process as much as possible, the concept of non-interactive audit is introduced. For the sake of ensuring the correctness of the audit, the reward pool mechanism is adopted, and the verification node can obtain reasonable rewards.
3) An analysis of the security of the scheme shows that it can achieve the expected security objectives. Numerous experiments on the ether block chain also showed the efficiency and effectiveness of the scheme.

This paper is organized as follows: Related work is presented in Chapter 2. The system model and design objectives are described in Chapter 3. The detailed description of the scheme is in Chapter 4. Chapter 5 contains an analysis of the security system. Chapter 6 discusses the performance of this experimental method. Chapter 7 is the conclusion of the paper.

II. RELATED WORK

A. BLOCKCHAIN

In view of the decentralization, tamper-proofness and traceability of blockchain technology, some blockchain-based data integrity auditing methods have been explored [18], [19]. Fan *et al.* zhe'ge [11] replaced the TPA with a smart contract, and the user signed an agreement with the CSP to prevent one party from denying it.. The data owner obtains the hash of the remote data through the block identifier and compares it with the hash value previously stored in the blockchain ledger. Obviously, this scheme cannot resist the replay attack carried out by the CSP. Yu *et al.* [13] decentralize the data without any TPA in their scheme. Their solution is effective against replay attacks due to the random challenge set generated in each audit request. To defend against dishonest provers and verifiers, Xu *et al.* [20] proposed an arbitrable data audit protocol that supports exchange hashing. Existing cloud storage service providers (CSP) may not have a fair compensation for users even if they damage data, and CSP may store redundant and duplicate data. Yuan *et al.* [24] proposed a deduplication scheme with public audit and fair arbitration.

B. BLOCKCHAIN EXPANSION

Although the emergence of blockchain has many advantages in data integrity auditing, with the increase of the number of users, the transaction throughput of the blockchain system will be seriously insufficient, and the storage burden on the blockchain is bound to increase. To address the above issues, the authors of conduct an extensive classification and comparison of blockchain scalability solutions [23], [25]. Zhou *et al.* [26] proposed a solution for blockchain scalability. The existing expansion schemes are designed to improve different layers, and are divided into layer-0 expansion, on-chain expansion, and off-chain expansion. Among them, on-chain expansion improves the efficiency of the blockchain by changing the basic protocol. Off-chain expansion does not change the basic protocol, and changes are made at the application layer to improve scalability. Layer-0 expansion improves blockchain scalability by changing the underlying data transmission protocol of the blockchain. The on-chain expansion scheme includes data layer improvement scheme, consensus layer improvement scheme and network layer improvement scheme. The basic idea is to increase the block size (either directly or indirectly) or reduce the block verification propagation time and consensus formation time. The off-chain expansion scheme mainly includes four methods: state channel, side chain, cross-chain and off-chain computation. The idea is to transfer some on-chain transactions to off-chain for execution, in order to reduce the processing pressure on the chain and improve the overall efficiency. While improving the performance of the blockchain, the off-chain scaling technology takes into account decentralization and security, and has various excellent properties.

III. MODEL AND SAFETY GOALS

A. SYSTEM MODEL

Based on the blockchain expansion technology, we propose a new audit scheme. Our scheme consists of three entities: data owner, cloud service provider and verifier. The system model is shown in Figure 1.

Data Owner (DO): The owner of the data, who can authorize other users to access and use the data.

Cloud Service Provider (CSP): Generally composed of multiple servers. It can provide users with massive data storage service.

Verifier: Audit the proof provided by the CSP and inform the DO of the result. In this scheme, the smart contracts

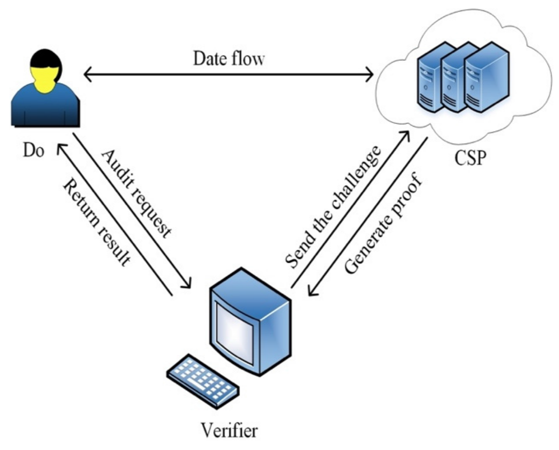


FIGURE 1. System model.

deployed on the blockchain and consensus nodes cooperate to perform audit tasks. In order to avoid affecting the user experience due to the communication with the CSP during the audit process as much as possible, the concept of non-interactive audit is introduced. The vulnerability to replacing attack and replay attack is also mitigated in this scheme.

B. SECURITY DEFINITION (SOUNDNESS)

The soundness of the scheme is proved by the following game between adversary A and challenger C:

1) C calls key generation algorithm $KeyGen(1^k)$ to generate keypair (pk, sk) and gives pk to A.
2) A tries to get a signature set $\Phi \leftarrow SigGen(F, pk, sk)$ by doing multiple interactions and queries with C
3) A outputs the audit proof P of the file F, the signature set Φ, and the current state τ.

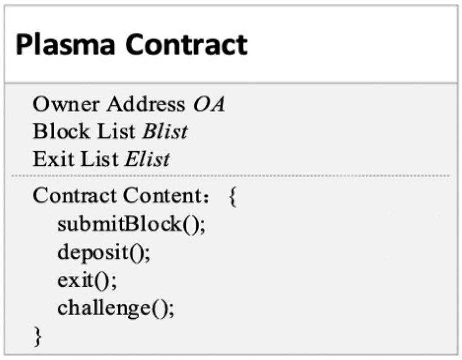
Let the probability of A forging a proof and passing the verification be $\mathrm{Adv}_{\mathrm{A}} = \Pr[\mathrm{Verify}\ (\mathrm{pk}, \mathrm{P}) = 1]$. The definition of adversary winning is that $\mathrm{Adv}_{\mathrm{A}}$ is non-negligible.

Definition 1: The proposed scheme is sound if there exists an efficient extraction algorithm. Adversary A outputs proof P based on file F, state τ and signature Φ. If the adversary A can win the game with a non-negligible probability, then there is an extraction algorithm that can recover the file F based on the signature Φ and the proof P, i.e.$Extract\ (pk, P, \Phi) = F$.

C. DESIGN GOALS

Our program should achieve the following goals:

4) Correctness: For all key pairs $(pk, sk) \leftarrow KeyGen(1^k)$, and for file F, state τ, the verification algorithm outputs $1 \leftarrow Verify\,(pk, ProofGen(sigGen\,(F, sk)\,, F, \tau))$.
5) Soundness: For any forged proof, it cannot pass the verification with a non-negligible probability.
6) Batch auditing: including multi-user single-task auditing and multi-user multi-task auditing. This is to ensure the efficiency of auditing.
7) Non-interactive: Reduce the number of interactions between the CSP and the user during the audit process.


Plasma Contract

Owner Address *OA*
Block List *Blist*
Exit List *Elist*

Contract Content: {
submitBlock();
deposit();
exit();
challenge();
}

FIGURE 2. Plasma Contract.

8) Public auditing: Ensure that any user including the data owner can challenge the CSP to verify the integrity of the data based on the certificate generated by the CSP.

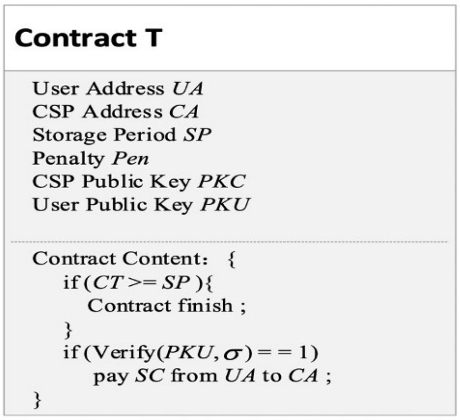
IV. SCHEME CONSTRUCTION

A. MAIN IDEA

In the existing data integrity audit schemes based on blockchain, an independent block is often created for each uploaded file for storage [14]. However, as the number of users of CSP services and the amount of data uploaded by users continue to increase, the cost of executing contracts and creating new blocks will continue to grow, reducing the effectiveness of the solution. At the same time, since a block is created separately for each file and the audit algorithm is written into it, the signatures cannot be aggregated, so batch auditing is not possible, resulting in a much lower audit efficiency than traditional solutions. For the problem of high cost for blockchain network maintenance and for user creation of new blocks caused by excessive block growth, a data integrity audit scheme based on blockchain expansion technology is proposed, and a reward pool mechanism is introduced to ensure that verification nodes can get a fair reward. The main idea is that the user first deploys the Plasma contract and the contract T to the main chain and sub-chain respectively. Then the user uploads the data and signature to the cloud. The data owner executes the *KeyGen* algorithm and the *SigGen* algorithm, and uploads the file F and the corresponding signature set Φ to the CSP. At the same time, the data owner deploys the Plasma contract (Figure 2) and the contract T (Figure 3) to the main chain and sub-chain respectively.

B. SMART CONTRACT BASED ON PLASMA EXPANSION TECHNOLOGY

Plasma is a blockchain expansion technology that has been widely researched and applied. By introducing this technology, the growth rate of blockchain can be slowed down and the storage and computational overhead can be reduced. The characteristics of smart contracts ensure that the blockchain network will automatically execute smart contract code, and the efficiency and accuracy of auditing can be



Contract T

```
User Address $UA$
CSP Address $CA$
Storage Period $SP$
Penalty $Pen$
CSP Public Key $PKC$
User Public Key $PKU$

Contract Content: {
    if($CT >= SP$){
        Contract finish ;
    }
    if(Verify($PKU, \sigma$) == 1)
        pay $SC$ from $UA$ to $CA$ ;
}
```

FIGURE 3. Contract T.

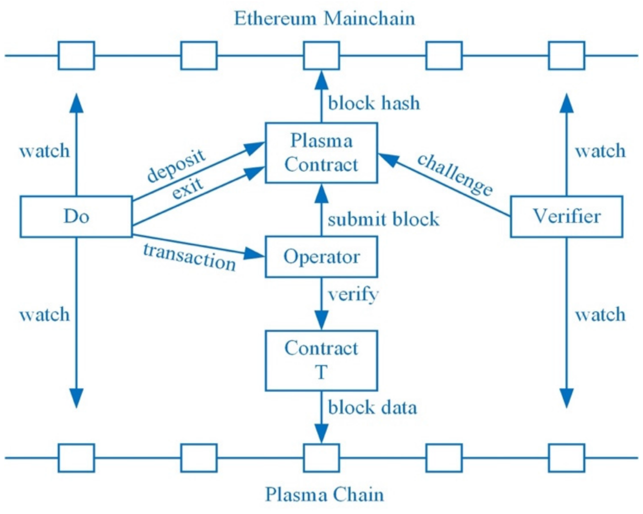


FIGURE 4. Plasma cycle.

guaranteed through smart contract. The entire life cycle of plasma is shown in Figure 4.

The contract includes 5 entities:

1) Main chain: responsible for recording the user's status information, sub-chain ID and information that proves the aggregation.
2) Plasma chain (child chain): Responsible for recording audit results. After the user registers on the main chain, a new Plasma chain is generated.
3) Plasma contract: It is the medium of communication between the main chain and the sub-chain. Plasma contracts process blocks submitted by subchains and store the hashes of the blocks on the main chain. In addition, it includes four algorithms: Deposit, verify, exit, and challenge.
4) Operator: Responsible for processing all transactions on the sub-chain, packaging them into blocks and storing them on the sub-chain, periodically submitting blocks to the Plasma contract, and submitting the state (hash value of the block) on the sub-chain to the main chain.
5) On-chain Wallet: Responsible for recording the status of subchains for users.

The data structure of the Plasma contract includes:

1) The owner of the contract: (i.e., the sender of the deployment contract transaction) address, that is, the sender of the deployment contract transaction.
2) Block list: Each Plasma block stores the Merkle root of the block and the time of block submission.
3) Exit list: The list of submitted exit applications. Each exit application records the address of the applicant and the location of the UTXO that applied for exit.

There are five main functions in the Plasma contract:

1) Submit Block: submit algorithm. Submit a block to the main chain.
2) Deposit (): The pre-storage algorithm. The new user executes this algorithm to generate a block containing only one transaction to record the number of tokens transferred from the main chain to the sub-chain.
3) Verify (): Verification algorithm. The smart contract executes this algorithm to verify the auditing proof.
4) Exit (): Exit algorithm. The user executes this algorithm to send an exit request to the Plasma contract. An "exit bond" is included in the application. If this exit request is successfully challenged, the withdrawal operation will be cancelled and the exit deposit will be sent to the challenging user.
5) Challenge (): Each exit request has a dispute period during which validators can challenge the exit in progress. A dispute is executed by calling the challenge () function. After the dispute is successful, the ongoing withdrawal operation will be canceled, and the deposit frozen by submitting the withdrawal application will be sent to the challenge.

The deployment and execution of the contract includes the following 5 steps:

1) Contract deployment: DO and CSP jointly confirm the content of the contract and deploy the Plasma contract to the main chain. DO sends a certain amount of main chain Token to the Plasma contract through the deposit algorithm to join the Plasma Chain, and deploys the verification contract to the sub-chain.
2) Proof verification: The operator processes the contracts deployed by users on the Plasma Chain, executes the verify algorithm to get the audit results, and generates blocks for them to add to the chain.
3) State Commitment: The operator periodically submits the hash value of the block on the Plasma Chain to the main chain as a proof of the state update of the sub-chain.
4) Status Monitoring: Users monitor the update status through the On-chain wallet.
5) User exit: When the user executes the exit algorithm to request to exit the sub-chain, the verifier can challenge the request by executing the challenge algorithm; when the operator uploads the recent verification to the main

chain, the user or consensus node can also challenge the request.

C. AUDIT SCHEME

1) SETUP PHASE

1) $KeyGen\left(1^k\right) \rightarrow pk, sk$ The data owner chooses the hash function $H(\cdot){:}\{0,1\}^* \rightarrow G$, and randomly chooses a signing key pair (ssk, spk). Then the user randomly selects $x \leftarrow Z_p, \mu \leftarrow G$ and calculates $h = g^x$. Private key is $sk{=}(ssk, x)$, and public key is $pk = (spk, g, h, e(a,b), H(\cdot))$.
2) $sigGen(F, pk, sk) \rightarrow \Phi$: Taking file F as an example, first divide the file into n blocks, i.e., $F = \{m_i\}_{1\leq i\leq n}$, where $m_i \in Z_p$ is the identifying name of the file F selected by the data owner. Signature set is $\Phi = \{\sigma_i\}_{1\leq i\leq n}$. In order to ensure the correctness of the file name, use the signature private key ssk to sign the name, and then generate a tag $t = name||Sig_{ssk}(name)$ for F. Finally, the data owner uploads the file F, data signature Φ and tag t to the cloud.

2) AUDIT PHASE

1) $ProofGen(\Phi, pk, \tau) \rightarrow \sigma$: In the input, τ is the time when this audit was initiated. First, the blockchain network as the auditor parses the tag t, and then verifies the correctness of the tag through the signature public key spk,and abort verification if it fails. Otherwise, the file identifier $name$ is reparsed from t. Then the auditor selects a set $I = \{1, 2, \ldots, c\}$ containing c elements, and selects a random number v_i for each element i in I, and the random number in $\{v_i\}_{i\in\{1,c\}}$ is substituted into the time hash function $H_2(v_i, timestamp, headerhash) \rightarrow Z_p^*$, which gives us the set $\{s_i\}_{i\in\{1,c\}}$. Then the auditor calculates $\sigma = \prod_{i\in\{1,c\}} \sigma_i^{s_i}$ and sends it to the blockchain network.
2) $ProofVerify$: $(\Phi, \sigma, pk) \rightarrow (0,1)$: After the blockchain network receives the auditing proof, the operator executes the verification algorithm stored on the sub-chain and verifies the following equation:

$$e(\sigma, g) = e(\prod_{i\epsilon[1,c]} H(name||i)^{s_i} \cdot \mu^{\lambda}, h) \quad (1)$$

If the verification is passed, the audit information and results will be saved to the sub-chain; otherwise, the data owner and the cloud should be notified that the verification fails, and the cloud will be punished.

V. SECURITY ANALYSIS

A. CORRECTNESS

If the CSP correctly stores the user's data, the *proof* generated by it can be verified by the *ProofVerify* algorithm. Equation (1) is verified as follows:

$$\begin{aligned}
&e(\sigma, g)\\
&\quad = e(\prod_{i\epsilon\,[1,c]} \sigma_i^{s_i}, g)\\
&\quad = e(\prod_{i\epsilon[1,c]} (H(name||i)\cdot\mu^{m_i})^{x\cdot s_i}, g)\\
&\quad = e(\prod_{i\epsilon[1,c]} (H(name||i)\cdot\mu^{m_i})^{s_i}, h)\\
&\quad = e(\prod_{i\epsilon[1,c]} H(name||i)^{s_i}\cdot\mu^{\lambda}, h)\\
&\quad = e(\prod_{i\epsilon[1,c]} H(name||i)^{H_2(v_i,sp,header)}\cdot\mu^{\lambda}, h)
\end{aligned}$$

B. SOUNDNESS

1) Anti-replacing attack: During the auditing process, if the CSP does not store the holder's data correctly, the signature generated by the non-challenge block data cannot be verified by the auditor.
2) Anti-replay attack: similar to anti-replacing attack. During the auditing process, if the CSP does not store the holder's data correctly, the signature generated by the CSP using the previous data block information cannot be verified by the auditor.

Proof:

1) The CSP forges signatures using non-challenge blocks in an attempt to pass verification. Assuming that the CSP uses a non-challenge block to forge the signature σ^-. If $e\left(\sigma^-, g\right) = e(\sigma, g)$ (2) holds, the replacing attack works. The proof is given below: Substitute the forged signature $\sigma^- = \prod_{i\epsilon[1,j-1]\cup[j+1,c]} \sigma_i^{s_i} + \sigma_{j^-}^{s_j}$ into equation (1), and get $e(\sigma^-, g) = e(\prod_{i\epsilon[1,j-1]\cup[j+1,c]} (H(name||i)\cdot\mu^{m_i})^{s_i}, h)\cdot e((H(name||j^-)\cdot\mu^{m_{j^-}})^{s_j}, h)$. For equation (1) to hold, equation (2) should hold. We have $e((H(name||j^-)\cdot\mu^{m_{j^-}})^{s_j}, h) = e\left((H(name||j)\cdot\mu^{m_j})^{s_j}, h\right)$, i.e., when $\sigma_{j^-}^{s_{j^-}} = \sigma_j^{s_j}$ (2) is satisfied, Equation (1) holds. Under the assumption of random oracle, equation (2) holds only when $H(name| = H(name||j^-)$ and $m_{j^-} = m_j$, which contradicts the previous assumption. Therefore, signatures forged by CSP using non-challenge blocks cannot be verified by auditors.
2) The CSP uses the information of previous challenge blocks to forge the signature and try to pass the verification. Assuming that the CSP uses previous challenge blocks' information to forge the signature σ^*, if $e\left(\sigma^-, g\right) = e(\sigma, g)$ (3) holds, the replay attack works. The proof is similar to the anti-replacing attack, and will not be repeated here.

C. BATCH AUDITING

There are usually two cases for batch auditing:

1) Multi-user single-task, that is, the auditor handles a single audit request from multiple users at the same time;
2) Multi-user multi-task, that is, the auditor handles multiple audit requests from multiple users at the same time.

TABLE 1. Comparison of data auditing schemes.

Schemes	Public Verifiability	Batch Audting	Fair Arbitration	Expansion Support
[24]	Yes	Yes	Yes	No
[27]	Yes	No	No	No
[22]	Yes	Yes	No	No
[20]	Yes	Yes	Yes	No
Our	Yes	Yes	Yes	Yes

TABLE 2. Computational analysis.

Schemes	Computational overhead in the integrity verification phase
[24]	2Map+(n+3)Exp+nMul
[27]	Map+(n+2)Exp+(n+1)Mul
[22]	Map+(n+2)Exp+(n+1)Mul
[20]	Map+(n+1)Exp+(n+2)Mul
Our	Map+(n+1)Exp+nMul

Proof:

1) If $\prod_{u_k\epsilon[1,k]} e(\sigma, g) = \prod_{u_k\epsilon[1,k]} e(\prod_{i\epsilon[1,c]} H(name||i)^{s_i} \cdot \mu^{\lambda}, h)$ (4) holds, the proposed scheme supports multi-user single-task auditing. Multi-user single-task auditing is a special case of multi-user multi-task auditing, so a detailed proof will be given in multi-user multi-task batch auditing.
2) If $\prod_{f_s\epsilon[1,s]} \prod_{u_k\epsilon[1,k]} e(\sigma, g) = \prod_{f_s\epsilon[1,s]} \prod_{u_k\epsilon[1,k]} e(\prod_{i\epsilon[1,c]} H(name||i)^{s_i} \cdot \mu^{\lambda}, h)$ (5) holds, the proposed scheme supports multi-user multi-task auditing. The proof is as follows:

$$\begin{aligned}
&\prod_{u_s\epsilon[1,s]} \prod_{f_k\epsilon[1,k]} e(\sigma, g)\\
&= \prod_{u_s\epsilon[1,s]} \prod_{f_k\epsilon[1,k]} e(\prod_{i\epsilon[1,c]} \sigma_i^{s_i}, g)\\
&= \prod_{u_s\epsilon[1,s]} \prod_{f_k\epsilon[1,k]} e(\prod_{i\epsilon[1,c]} (H(name||i)\cdot\mu^{m_i})^{x\cdot s_i}, g)\\
&= \prod_{u_s\epsilon[1,s]} \prod_{f_k\epsilon[1,k]} e(\prod_{i\epsilon[1,c]} H(name||i)^{s_i}\cdot\mu^{\lambda}, h)\\
&= \prod_{u_s\epsilon[1,s]} \prod_{f_k\epsilon[1,k]} e(\prod_{i\epsilon[1,c]} H(name||i)^{H_2(v_i,sp,header)} \cdot \mu^{\lambda}, h)
\end{aligned}$$

D. PERFORMANCE COMPARISON

Table 1 shows the comparison between the proposed scheme and scheme in [24], scheme in [27], scheme in [22], and scheme in [20] in four aspects: public verification, batch auditing, fair arbitration, and blockchain expansion. All schemes support public auditing. The schemes of in [24] and in [20], and our proposed scheme all use the blockchain network as the auditor, so they all support fair arbitration. As shown in Table 2, our scheme is the only solution that supports all four aspects.

Table 2 shows the formal computational costs of schemes in [24], [27], [22], [20] and our proposed scheme. Map represents a bilinear pairing operation. Mul represents a multiplication operation on an elliptic curve. Indicates the number of challenged data blocks. Exp represents an exponentiation operation on an elliptic curve.

VI. PERFORMANCE ANALYSIS

A. COMPUTATIONAL OVERHEAD

Experiments are conducted using (JPBC) and Solidity and under Intel Core i9-9800HK CPU 2.3GHz 32RAM Linux environment. The experimental results of our proposed scheme are compared with those of schemes in [20], [22], [24], [27]. Since the I/O delay mainly depends on factors such as hardware conditions and scheduling algorithms, and has little to do with the scheme itself, the experiment mainly tests the signature generation time, proof generation time and gas cost of integrity verifying. Proof generation time is the time the CSP uses to produce proof of integrity verification. Integrity verification Gas cost is the cost of the smart contract for integrity verification results. Figure 5 shows the comparison of the signature generation time between our proposed scheme and that of scheme in [20], [22], [24], [27]. In order to be closer to the actual use case, the file block is divided into n data blocks, where n ranges from 100 to 1000, and the data block size is 5MB.

It can be seen from the figure that when the number of data blocks increases from 100 to 1000, the time of signature generation of our proposed scheme increases from 0.41s to 4.14s; the time of scheme [24] increases from 0.54s to 6.3s; the time of scheme [20] increases from 1.55s to 8.01s; the time of scheme [22] increases from 2.34s to 20.55s; the time of scheme [27] increases from 3.51s to24.94s.

Figure 6 shows a comparison of the proof generation time between our proposed scheme and that of scheme in [20], [22], [24] and [27].

In order to make the experimental results more accurate, the number of challenge blocks is selected from 100 to 1000 in the experiment. It can be seen from the figure that when the number of challenge blocks increases from 100 to 1000, the audit proof generation time of

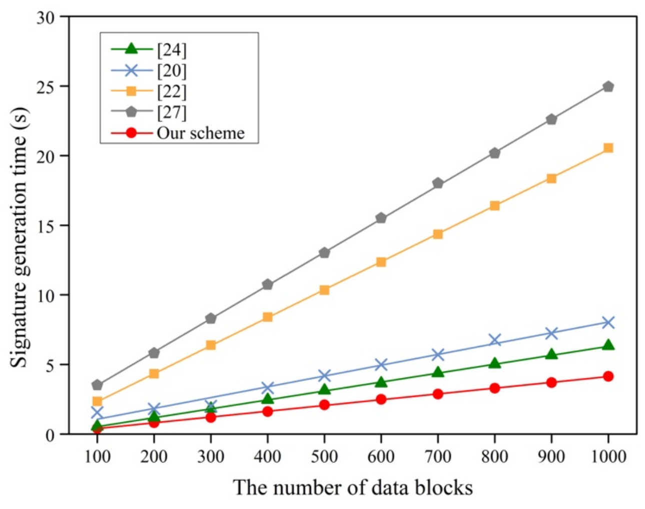


FIGURE 5. Comparison of signature generation time.

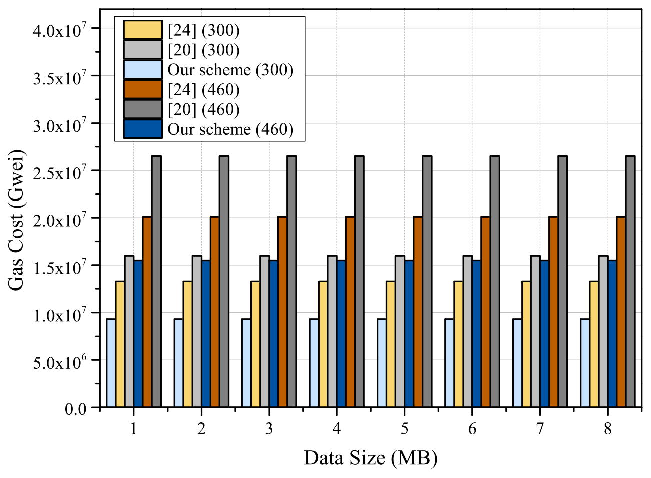


FIGURE 7. Gas cost of integrity verifying.

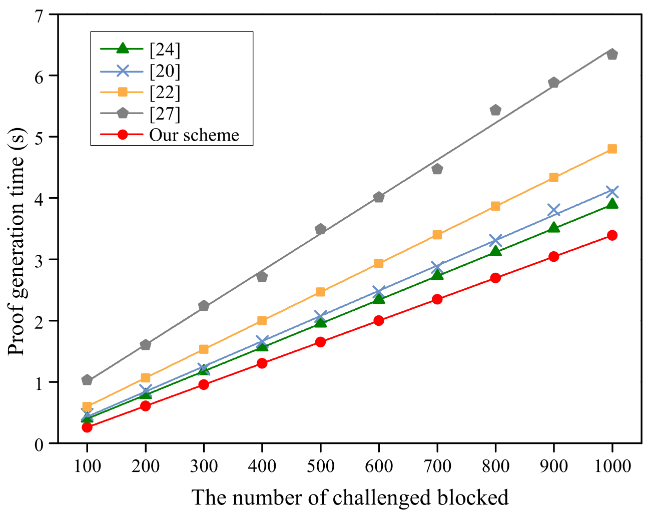


FIGURE 6. Comparison of proof generation time.

our proposed scheme increases from 0.26s to 3.4s; the time of scheme [24] increases from 0.4s to 3.9s; the time of scheme [20] increases from 0.47s to 4.1s; the time of scheme [22] increases from 0.6s to 4.8s; the time of scheme [27] increases from 1s to 6.3s.

Figure 7 shows the comparison of gas cost for integrity verification between our proposed scheme, scheme [24], and scheme [20]. Gas represents the amount of resources that must be consumed and cannot be refunded in the Ethernet system. EVM must consume gas when executing code. Therefore, gas reflects the amount of resources consumed by EVM when working. All EVM transactions and smart contract executions require gas fees. In our scheme, most operations are performed on the blockchain, so the gas consumption reflects the cost of the scheme. According to the standard of the Ethereum Association, we set the gas price to 2.5 GWei (the basic currency unit of Ethereum, 109 GWei = 1 ether) in our experiments.

The experiment uses Solidity to program smart contracts and experiment on the Ethereum blockchain. Since our solution only stores the user's state information, sub-chain ID and proof aggregation information on the main chain, the gas cost of our scheme is much lower than that of scheme [24] and scheme [20]. We set the number of challenge blocks to 300 and 400, and data size ranges from 1MB to 10MB. The results of the performance evaluation are the average of 20 experiments. It can be seen from Figure 7 that when the number of randomly selected data blocks is constant, the gas cost does not grow with the increase of the data size. When validating 300 randomly selected data blocks, the gas consumption value is 0.93 × 107 Gwei (0.013 ether), and when validating 460 randomly selected data blocks, the gas consumption value is 1.55 × 107 Gwei (0.020 ether).

VII. CONCLUSION

As cloud computing and cloud storage technologies evolve faster and faster, the amount of data in cloud storage grows explosively, how can we ensure that the full information stored by users on cloud servers becomes an important topic for discussion. This article proposes a data integrity scheme based on block chain expansion technology. In our scheme, we use the blockchain network to overcome some of the shortcomings of traditional auditing, improving the efficiency and security of the scheme. In addition, we introduce plasma sub-chain and deploy smart contracts on the main chain and sub-chain respectively. Through this protocol, the storage pressure of the main chain can be greatly reduced, the growth rate can be slowed down, the storage and computational overhead can be reduced, and the system performance can be improved. At the same time, the reward pool mechanism and the concept of non-interactive audit are introduced to ensure the correctness of the audit and avoid the interaction between the smart contract platform and the CSP during the contract execution process, and the solution can achieve the expected security goals.

REFERENCES

[1] K. Hao, J. Xin, Z. Wang, and G. Wang, "Outsourced data integrity verification based on blockchain in untrusted environment," *World Wide Web*, vol. 23, no. 4, pp. 2215–2238, Jul. 2020.

[2] Y. Fan, X. Lin, G. Tan, Y. Zhang, W. Dong, and J. Lei, "One secure data integrity verification scheme for cloud storage," *Future Gener. Comput. Syst.*, vol. 96, pp. 376–385, Jul. 2019.

[3] H. Wang and J. Zhang, "Blockchain based data integrity verification for large-scale IoT data," *IEEE Access*, vol. 7, pp. 164996–165006, 2019.

[4] Z. Miao, C. Ye, P. Yang, R. Liu, B. Liu, and Y. Chen, "A scheme for electronic evidence sharing based on blockchain and proxy re-encryption," in *Proc. 4th Int. Conf. Blockchain Technol. Appl.*, Dec. 2021, pp. 11–16.

[5] F. Kefeng, L. Fei, Y. Haiyang, and Y. Zhen, "A blockchain-based flexible data auditing scheme for the cloud service," *Chin. J. Electron.*, vol. 30, no. 6, pp. 1159–1166, Nov. 2021.

[6] K. He, J. Shi, C. Huang, and X. Hu, "Blockchain based data integrity verification for cloud storage with T-Merkle tree," in *Proc. Int. Conf. Algorithms Archit. Parallel Process.* Cham, Switzerland: Springer, Oct. 2020, pp. 65–80.

[7] Y. Lei, Z. Jia, Y. Yang, Y. Cheng, and J. Fu, "A cloud data access authorization update scheme based on blockchain," in *Proc. 3rd Int. Conf. Smart BlockChain (SmartBlock)*, Oct. 2020, pp. 33–38.

[8] Y. Yuan, J. Zhang, W. Xu, and Z. Li, "Identity-based public data integrity verification scheme in cloud storage system via blockchain," *J. Supercomput.*, vol. 78, pp. 8509–8530, Jan. 2022.

[9] S. Wang, D. Zhang, and Y. Zhang, "Blockchain-based personal health records sharing scheme with data integrity verifiable," *IEEE Access*, vol. 7, pp. 102887–102901, 2019.

[10] A. Liu, Y. Wang, and X. Wang, "Blockchain-based data-driven smart customization," in *Data-Driven Engineering Design*. Cham, Switzerland: Springer, 2022, pp. 89–107.

[11] K. Dhyani, J. Mishra, S. Paladhi, and I. S. Thaseen, "A blockchain-based document verification system for employers," in *Proc. Int. Conf. Comput. Intell. Data Eng.* Singapore: Springer, 2022, pp. 123–137.

[12] K. Xu, W. Chen, and Y. Zhang, "Blockchain-based integrity verification of data migration in multi-cloud storage," *J. Phys., Conf. Ser.*, vol. 2132, no. 1, Dec. 2021, Art. no. 012031.

[13] G. Xu, S. Han, Y. Bai, X. Feng, and Y. Gan, "Data tag replacement algorithm for data integrity verification in cloud storage," *Comput. Secur.*, vol. 103, Apr. 2021, Art. no. 102205.

[14] G. Xie, Y. Liu, G. Xin, and Q. Yang, "Blockchain-based cloud data integrity verification scheme with high efficiency," *Secur. Commun. Netw.*, vol. 2021, pp. 1–15, Apr. 2021.

[15] U. Arjun and S. Vinay, "Outsourced auditing with data integrity verification scheme (OA-DIV) and dynamic operations for cloud data with multi-copies," *EAI Endorsed Trans. Cloud Syst.*, vol. 7, no. 20, Jul. 2018, Art. no. 169423.

[16] A. V. Ezhil, G. K. Indra, and K. Kulothungan, "Auditable attribute-based data access control using blockchain in cloud storage," *J. Supercomput.*, vol. 78, pp. 10772–10798, Jan. 2022.

[17] R. Mishra, D. Ramesh, D. R. Edla, and M. C. Trivedi, "Blockchain assisted privacy-preserving public auditable model for cloud environment with efficient user revocation," *Cluster Comput.*, pp. 1–25, Jan. 2022.

[18] X. Tao, Y. Liu, P. K.-Y. Wong, K. Chen, M. Das, and J. C. P. Cheng, "Confidentiality-minded framework for blockchain-based BIM design collaboration," *Autom. Construct.*, vol. 136, Apr. 2022, Art. no. 104172.

[19] P. Wei, D. Wang, Y. Zhao, S. K. S. Tyagi, and N. Kumar, "Blockchain data-based cloud data integrity protection mechanism," *Future Gener. Comput. Syst.*, vol. 102, pp. 902–911, Jan. 2020.

[20] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Trans. Services Comput.*, vol. 13, no. 2, pp. 289–300, Mar. 2020.

[21] 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, 2019.

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

[23] L. Chen, Q. Fu, Y. Mu, L. Zeng, F. Rezaeibagha, and M.-S. Hwang, "Blockchain-based random auditor committee for integrity verification," *Future Gener. Comput. Syst.*, vol. 131, pp. 183–193, Jun. 2022.

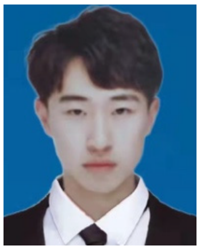
[24] H. Yuan, X. Chen, J. Wang, J. Yuan, H. Yan, and W. Susilo, "Blockchain-based public auditing and secure deduplication with fair arbitration," *Inf. Sci.*, vol. 541, pp. 409–425, Dec. 2020.

[25] C. Yang, Y. Liu, F. Zhao, and S. Zhang, "Provable data deletion from efficient data integrity auditing and insertion in cloud storage," *Comput. Standards Interface*, vol. 82, Aug. 2022, Art. no. 103629.

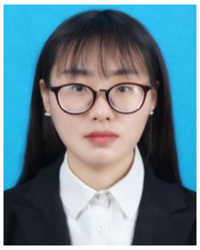
[26] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to scalability of blockchain: A survey," *IEEE Access*, vol. 8, pp. 16440–16455, 2020.

[27] Y. Li, Y. Yu, G. Min, W. Susilo, J. Ni, and K.-K.-R. Choo, "Fuzzy identity-based data integrity auditing for reliable cloud storage systems," *IEEE Trans. Dependable Secure Comput.*, vol. 16, no. 1, pp. 72–83, Jan. 2019.

ZHENPENG LIU received the B.S. and M.S. degrees from the School of Cyberspace Security and Computer, Hebei University, and the Ph.D. degree from Tianjin University. He is currently a Professor with the School of Cyberspace Security and Computer, Hebei University. His research interests include big data, cloud computing, and information security research.



YONGJIANG FENG is currently pursuing the M.S. degree with the School of Cyberspace Security and Computer, Hebei University, Baoding, Hebei. His research interests include cloud computing, cloud storage, and data integrity verification.



LELE REN is currently pursuing the M.S. degree with the School of Cyberspace Security and Computer, Hebei University, Baoding, Hebei. His research interests include cloud computing security, privacy protection, and data integrity verification.

WEIHUA ZHENG received the M.S. degree from the Hebei University of Engineering. He is currently an Associate Professor with the School of Information Engineering, Handan University, and the School of Information Science and Engineering, Xinjiang University of Science and Technology. His research interests include computer application, big data, cloud computing, and information security research.