sChain: An Efficient and Secure Solution for Improving Blockchain Storage

Lipeng Wang, Zhi Guan, Zhong Chen, and Mingsheng Hu

***Abstract*— Emerging blockchain technology has become the cornerstone of many applications providing trusted data services. However, existing blockchain platforms cannot meet the growing demand for big data storage. Blockchain should duplicate both transactions and other user-defined data across nodes for integrity assurance. The rapid expansion of data on blockchain (on-chain data) increases the difficulty of deploying a full node, resulting in decreasing the degree of decentralization and adding the risk of broken data. To tackle these problems, we propose sChain, a novel framework for improving blockchain storage capacity, which does not revise blockchain implementation and can be applied to almost all the existing blockchain platforms. sChain outsources the user data to storage devices that are structurally external to the blockchain network. In theory, a user can outsource unlimited data to sChain. However, those off-chain data may suffer from corruption. To verify the data integrity, we propose a new provable data possession (PDP) scheme, which does not need a centralized entity to maintain any secret keys and therefore eliminates a single point of failure. What is more, we also design a prototype to accelerate the proposed PDP scheme through Intel SGX technology and parallel processing. Security analysis and evaluation results show that sChain can protect data security and effectively improve the blockchain storage capacity, respectively.**

***Index Terms*— Blockchain, data integrity, off-chain storage, provable data possession, scalability.**

I. Introduction

BLOCKCHAIN is a digitally distributed, decentralized ledger and noteworthy in its use with digital currencies [1]. In July 2020, the Bitcoin market cap reached $195.8 billion, accounting for 65% of the entire cryptocurrency market [2]. Without a trusted third-party center, blockchain can provide credible data services. In recent years, blockchain has become a hot research topic in both academia and industry due to its extensive application prospects. However, existing blockchain platforms cannot meet the growing demand for big data storage. Blockchain should duplicate both transactions and other user-defined data across nodes for integrity assurance, resulting in on-chain storage expensive. Huge storage consumption places significant burdens on blockchain and limits its scalability. For example, as of July 2021, the amount of data stored on Bitcoin has exceeded 350 GB and is still growing at a rate of 58 GB per year. Meanwhile, the data stored on Ethereum reached over 7 TB [3]. The rapid expansion of outsourcing data on blockchain increases the difficulty of deploying a full node, resulting in decreasing the degree of decentralization and adding the risk of data corruption.

How to efficiently improve the storage capacity of blockchain while protecting data security is the main focus of this paper. Some solutions have been proposed to address these aforementioned challenges recently, such as Swarm and StorJ [4]. The common idea is to store the user data on off-chain nodes and the corresponding metadata on blockchain nodes and ensure that the off-chain data are consistent with the on-chain data. Utilizing the method, in theory, the user can outsource unlimited data to blockchain. However, due to natural disasters, power outages, equipment failures, chemical corrosion and cyberattacks, the off-chain data may be corrupted. In addition, most of existing blockchain scaling solutions are implemented with time-consuming cryptographic operations, which may be low efficient when to store the user data. How to guarantee security of the off-chain data with high efficiency has become the prerequisite for scaling blockchain.

To overcome the data security issue, a promising solution is to store hash values and metadata representing the off-chain data status and structure information on blockchain and then leverage them to verify the off-chain data integrity. If the off-chain data are checked as compromised, blockchain can take stop-loss measures in time. However, current audit schemes based on hash functions have several drawbacks. First, an auditor should retrieve the off-chain data through the network to calculate their corresponding hash values, which may cause high bandwidth costs. Second, if the off-chain storage holds a large file and its corresponding hash value, the verifier will wait for a long time to retrieve the entire file and then perform the data verification, which may incur a long time delay and limit its flexibility of customizing the auditing data volume. Third, if users revise their off-chain files, even if only by

Manuscript received 18 April 2022; revised 30 January 2023 and 1 May 2023; accepted 2 June 2023. Date of publication 13 June 2023; date of current version 26 June 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1005404, in part by the National Natural Science Foundation of China under Grant 62172010, and in part by the Henan Province Higher Education Key Research Project under Grant 22A520048. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Edgar Weippl. *(Corresponding author: Lipeng Wang.)*

Lipeng Wang is with the School of Information Science and Technology, Zhengzhou Normal University, Zhengzhou 450044, China, and also with the School of Computer Science, Peking University, Beijing 100871, China (e-mail: wlpscu@126.com).

Zhi Guan and Zhong Chen are with the School of Computer Science, Peking University, Beijing 100871, China (e-mail: guan@pku.edu.cn; zhongchen@pku.edu.cn).

Mingsheng Hu is with the School of Information Science and Technology, Zhengzhou Normal University, Zhengzhou 450044, China (e-mail: hero_jack@163.com).

Digital Object Identifier 10.1109/TIFS.2023.3285489

1556-6021 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

one bit, the system needs to recalculate their hash values, which results in high computational costs. Last, the third-party auditor (TPA) in charge of auditing the off-chain data may be compromised by an attacker, thereby tampering with the auditing results.

The provable data possession (PDP) algorithm proposed in 2007, which is regarded as an efficient data integrity auditing solution, can handle the aforementioned challenges. However, for an emerging PDP scheme, calculating data authenticators is always time-consuming. When fitting a PDP scheme to a blockchain, leveraging smart contracts to perform verification may introduce a long time delay. From the perspective of key management, current PDP schemes mainly include PKI-based, identity-based, and certificate-less schemes [5]. However, these schemes require a centralized authority to generate or distribute secret keys, which can lead to a single point of failure. Although some decentralized PDP schemes permit users to manage their own keys, the subsequent data verification procedure may fail if users lose their keys. Therefore, these PDP schemes cannot be directly applied to the decentralized architecture of blockchain without modification. To address this, we proposed a new PDP scheme that does not require any entity to maintain secret keys. This allows the user to generate the private key, upload files, and then delete the key.

To improve blockchain storage, we will leverage the proposed PDP scheme and Intel SGX to design a decentralized prototype, namely, sChain. The new framework will outsource the user data to off-chain storage that are structurally external to the blockchain network, and utilizes the proposed PDP scheme to verify the off-chain data integrity. The SGX technology proposed by Intel Corporation can build a trusted execution environment (TEE) and accelerate the PDP scheme with a built-in instruction set. Moreover, Intel SGX can help protect the secret key in use via application isolation technology. By protecting selected code and data from modification, the technology increases application security. What is more, we also accelerate sChain through Intel SGX technology and parallel processing. sChain does not revise blockchain implementation and can be applied to almost all the existing blockchain platforms. sChain does not need to store and maintain any secret keys. Optionally, sChain allows a user to choose a symmetric encryption algorithm and a secret key to encrypt his or her outsourcing data. Even if the user loses his or her secret key, sChain can still perform the data auditing normally.

The contributions are as follows:

1. We present sChain, a novel framework for improving blockchain storage capacity through Intel SGX technology and parallel processing, which does not need to revise blockchain implementation and can be applied to most blockchain platforms.

2. A new public PDP scheme is proposed, which can verify the off-chain data integrity for sChain with low communication and computation costs. The new scheme does not need to store and maintain any secret keys, which eliminates a single point of failure.

3. We analyze the security of the PDP scheme and evaluate the performance of sChain. Experimental results show that sChain can increase the storage capacity of blockchain with low time costs.

II. Related Work

When leveraging the off-chain storage to improve the capacity of blockchain, it is necessary to ensure that those outsourcing data are not maliciously tampered with or broken in their life cycles [4]. A common strategy is to verify the off-chain data integrity periodically with a hash-based scheme and leverage the auditing result to guide the system to take stop-loss measures in time. There are some related studies emerging. Storj is a P2P data storage system based on blockchain and compatible with the Amazon S3 protocol, which allows an owner to share his or her storage and bandwidth and be rewarded accordingly [6]. Storj stores hash values of the off-chain data on blockchain and makes use of them to check the data integrity. Ericsson can also provide data integrity services for the user's data [7]. The system utilizes the key-less signature infrastructure (KSI) technology to generate signatures and exploits hash trees along with timestamps to sign multiple documents at the same time. When verifying these signatures, the client should send them to a proxy server for comparison. However, auditing schemes based on hash functions have several drawbacks, such as high bandwidth consumption, inability to customize the amount of data to be audited, and high data revision costs. Later methods [8], [9], [10] utilize more efficient data structures (i.e., Merkle tree) to accelerate data verification and reduce the communication overhead. However, existing data auditing schemes based on Merkle tree would eventually run out of those fresh data blocks to be challenged and incur storage overhead to keep those hash values.

To solve the aforementioned problems, Atenese et al. first proposed the provable data possession algorithm based on homomorphic authenticators and the random sampling strategy, which can verify the data integrity without holding them [11]. The scheme can detect a certain percentage of corrupted data but is not able to recover them. In 2022, Dhakad et al. proposed a privacy-preserving PDP scheme for auditing cloud data, which also supports batch auditing, dynamic data operations [12]. Some other PDP schemes were also developed [13], [14], [15], [16], [17]. Later, Juels et al. proposed a proof of retrievability (POR) mode based on the pseudorandom sampling strategy and redundancy error correcting codes technology [18]. The PoR scheme can check and recover those broken data at untrusted servers with high probability. Shacham et al. proposed a compact PoR scheme based on BLS signatures, along with the formal security proofs against arbitrary adversaries [19]. In 2022, Yang et al. proposed an identity-based PoR scheme for cloud storage, which can recover these broken file blocks from outsourced data [20]. Some other PoR schemes were also proposed [21], [22], [23]. Both PDP and PoR leverage the "challenge-response" paradigm to avoid the transmission of auditing data and effectively reduce communication overhead.

Some PDP schemes have been proposed in the PKI model [24], [25], [26]; however, the certificate management

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

overhead is high. To overcome this issue, Wang et al. proposed an identity-based PDP scheme [27], in which a user leverages his or her identity as the public key and a trusted public key generator (PKG) to manage the private key. Although some identity-based PDP schemes have been proposed [28], [29], [30], [31], key escrow problems still remain. As a result, some certificate-less PDP schemes have been proposed [32], [33], but they require a key generation center to help generate the user keys. The above schemes necessitate the involvement of a centralized authority to generate or distribute secret keys, which can lead to a single point of failure. Therefore, they cannot be directly applied to the decentralized architecture of blockchain without modification. To overcome the problem, we will design a new PDP scheme that does not require any entity to store or maintain secret keys.

Considering the role of TPA, the emerging PDP algorithms can be divided into two categories: private PDP and public PDP. A private PDP scheme only permits a TPA who holds the private key of the data owner to perform the data verification. When the TPA and the data owner are undertaken by different entities, they must trust each other and share the secret key. A public PDP scheme allows any entity who holds the data owner's public key to verify data. Both the public and private PDP schemes can be applied to blockchain to verify user data. Wang et al. [34] proposed a blockchain-based private PDP scheme, which can realize client anonymity. In 2021, Francati et al. proposed Audita, a system for auditing off-chain data integrity [35]. In Audita, blockchain nodes are classified into two categories, namely, block-creators and storage nodes. Those block-creators participate in the consensus protocol and act as a TPA simultaneously, while storage nodes are responsible for storing the user data. The PDP scheme proposed in Audio supports public verification, so TPA does not need to hold the private key of a data owner. Audita relies on a third-party node called a "dealer" to distribute the file blocks, which may introduce a single point of failure. In 2022, a secure and efficient data auditing framework based on blockchain was proposed, which also provided a convenient data mitigation solution [36]. The framework needs to perform time-consuming bilinear pairing operations on blockchain during the verification procedure, which may incur high time cost. Even worse, most of existing standard cryptographic libraries that support bilinear pairing operations cannot be directly ported to blockchain without sacrificing performance. To overcome the problem, we will design a lightweight PDP scheme discarding bilinear pairing operations to verify the data integrity. IntegrityChain is another decentralized implementation of checking the integrity of those redundant backup off-chain data [37]. In IntegrityChain, the user should generate and maintain secret keys required by the PDP scheme. Once these keys are broken or lost, IntegrityChain is unavailable. In essence, IntegrityChain is still a centralized architecture from the perspective of the key management process. Du et al. proposed a scheme that utilizes smart contracts to audit off-chain data, which makes the data auditing operation transparent but may introduce a long time delay [38].

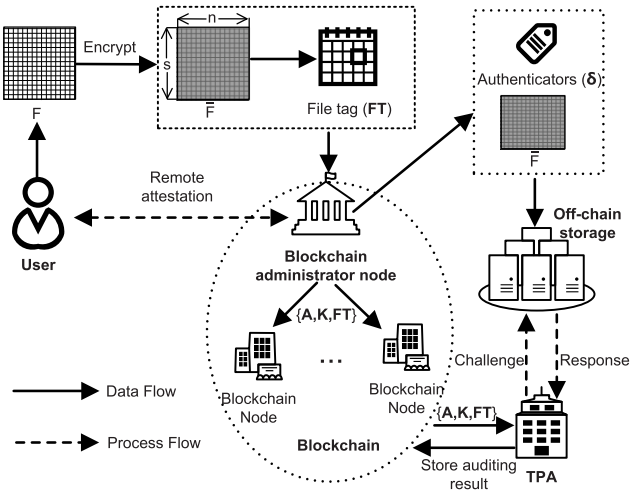


Fig. 1. System framework of sChain.

III. System Model

A. System Framework

Fig.1 demonstrates our proposed framework, sChain, which is designed to enhance the storage capability of blockchain. There are five different entities: user, blockchain, blockchain administrator node (BAN), TPA and the off-chain storage. The user owns a large amount of files, which will be outsourced to the off-chain storage. For privacy preservation, the user should encrypt an uploading file with a secret key and hold the key secretly. Then, the encrypted file is segmented into different blocks, and each block has a preset number of sectors. The user should calculate the corresponding file tag (FT) for the file. A blockchain administrator node is selected from the blockchain participants and should be supported by Intel SGX technology. The user and the administrator node run the remote attestation protocol provided by Intel SGX technology to build a secure channel between them. The protocol utilizes a modified sigma protocol to conduct a Diffie-Hellman Key Exchange (DHKE) between the user and the administrator node. The shared key obtained from this exchange allows the user to encrypt secrets to be provisioned to the administrator node. The administrator node's enclave is able to derive the same key and use it to decrypt the secrets [39]. The user sends the encrypted file and the file tag to the administrator node through the secure channel and entrusts the node to calculate file authenticators (δ) and public values {A,K}. Then, the encrypted file along with its authenticators will be outsourced to the off-chain storage, and the file tag along with those public values will be stored on blockchain. The TPA is responsible for verifying the data integrity of the off-chain storage and can be taken by the user or the blockchain administrator node. When performing integrity verification, the TPA first poses a challenge to off-chain storage. Then, a proof is generated as the response, which will be utilized by the TPA to verify the data integrity.

We now describe the five entities involved in our proposed framework.

1) User: The user owns a number of files and outsources them to the external storage of blockchain. Concerning data

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

privacy, those files can be encrypted with a secret key, and the key is retained by the user himself or herself. Note that the encryption step is optional. Even if the user loses the secret key, sChain can still perform the data auditing normally. Because user devices may vary largely in computing power and some of them cannot perform complex arithmetic calculations, sChain allows the user to assign these time-consuming operations to the blockchain administrator node, which can accelerate them in parallel. We will discuss this later.

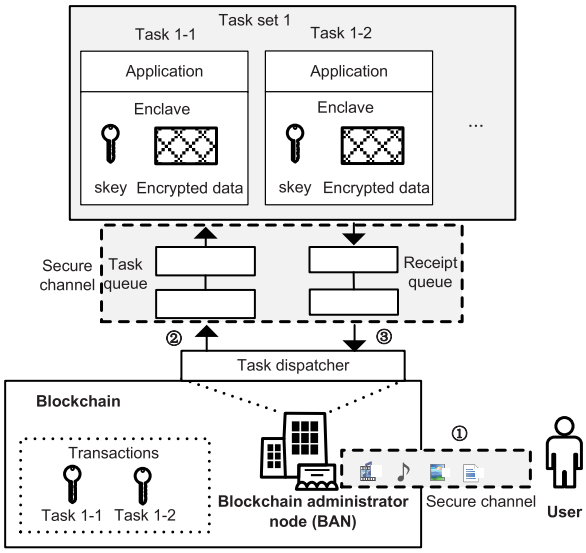
2) Blockchain: Blockchain is comprised of peers with the ability to validate and store transactions. The data on blockchain are tamper-resistant and tamper-evident. Although improper implementation of blockchain incurs many security issues, such as the 51% attack and the DAO attack, we assume the underlying blockchain platform for sChain is safe. For more details about blockchain security, please refer to [40].

3) Blockchain Administrator Node: A blockchain administrator node is selected from one of the participants in the blockchain network and powered by Intel SGX CPUs. Through the remote attestation protocol and a built-in security instruction set provided by Intel SGX, the user can trust the blockchain administrator node and delegate the task of computing file authenticators needed by the proposed PDP scheme to the node. Intel SGX technology, which aims to protect against both software and hardware attacks, can embed a trusted execution environment, which is organized as an enclave, in the application process. For safety reasons, the enclave should not store any data when the CPU stops running. Moreover, because the memory of an enclave is limited in size, a blockchain administrator node cannot process a large amount of data at a time.

4) TPA: The TPA is responsible for checking the data integrity of the off-chain storage. Since our proposed PDP scheme supports public auditing, any authorized entity can verify the data integrity. In our implementation, either the administrator node or the user can take the role of TPA and does not need to maintain any secret key.

5) Off-Chain Storage: In blockchain, the off-chain storage refers to the data repository residing external to the blockchain network [4]. Traditional data stores, such as DAS, NAS, centralized cloud storage and cloud computing, can be considered off-chain storage [3]. sChain should support secure data interaction between blockchain and the external storage and guarantees that the off-chain data cannot be tampered with. To achieve this objective, we design a new PDP scheme based on Intel SGX technology to verify the off-chain data integrity, which can ensure consistency between on-chain and off-chain data.

For a PDP scheme, when the user outsources a file, he or she should split it into fixed size blocks and then calculate file authenticators. However, the traditional implementation of retrieving and handling blocks one by one for the user is time-consuming. For example, when outsourcing a file with $1GB$ in size, the user can fix the block size as $4KB$ recommended in [3], and therefore the number of file blocks equals $1GB/4KB = (1024 \times 1024)KB/4KB = 262144$. Assuming calculating the file authenticator for a file block takes 1 ms, it will cost the user $262144 \times 1ms = 262s$ to process the whole file. The time cost is not acceptable for those time-sensitive applications. A promising solution is to delegate these time-consuming operations to the blockchain administrator node and perform them in parallel. To accelerate these operations, the blockchain administrator node should be provided with adequate computing resources. Users are allowed to remotely attest executing results with Intel SGX technology so that they can outsource their data to the off-chain storage with confidence.



① The user outsources files to BAN.
② BAN encapsulates the user's request of calculating file authenticators for the PDP scheme into a task, which is then pushed into the task queue.
③ BAN retrieves file authenticators from a task in the receipt queue, and outsources the files along with authenticators to the off-chain storage, the execution result to blockchain.

Fig. 2. File outsourcing framework.

B. File Outsourcing Framework

The main idea of our proposed framework for improving blockchain storage capacity is to migrate the on-chain data to the off-chain storage and ensure data consistency through a proposed PDP scheme. However, the task of computing file authenticators required by the PDP scheme is time-consuming. To improve efficiency, the involved data migrating operation is undertaken by the blockchain administrator node and executed in parallel. When a user outsources the data to blockchain, the blockchain administrator node will be entrusted to calculate file authenticators. Fig.2 demonstrates the proposed file outsourcing framework.

In the file outsourcing framework, the blockchain administrator node needs to be certified through the IAS or the DCAP protocol and therefore be trusted by the user. The node is responsible for participating in the blockchain consensus and calculating file authenticators. Upon receiving the user uploading request, the administrator node first selects a private-public key pair ($skey$, pky), which helps the user and the administrator node negotiate a session key $sskey$ by exploiting the remote attestation protocol provided by Intel SGX technology. With $sskey$, the subsequent data transmission between the blockchain administrator node and the user is encrypted, which can protect data from sniffing. The user

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

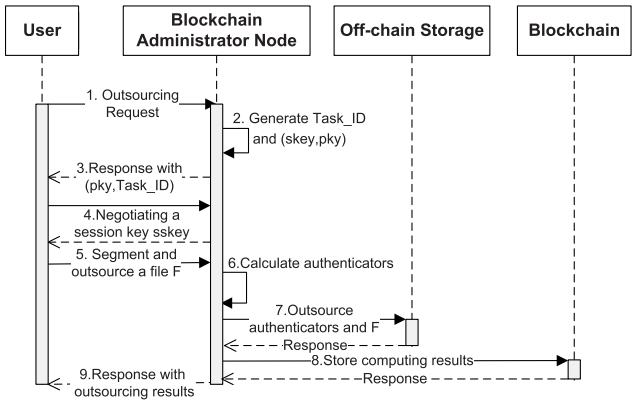


Fig. 3. Process of uploading the user file.

request and *pky* are encapsulated into a task, which is then pushed into the task queue. The blockchain administrator node can construct an index *Task_ID* by combining *pky* with some random numbers, which can uniquely identify a task. sChain should publicize *Task_ID* by issuing a blockchain transaction for postmortem audits. To improve the efficiency, a certain number of tasks retrieved from the task queue are organized into a task set and can be handled in parallel. Execution results are pushed into the receipt queue and returned to the user by the blockchain administrator node. It should be noted that when the user outsourcing process terminates, the key pair (*skey*, *pky*) becomes invalid and must be regenerated when a user issues a new request.

Fig.3 depicts a sequence diagram of outsourcing files to the off-chain storage, and the detailed descriptions are as follows.

1) The user selects a blockchain administrator node, and issues a request to it. Upon receiving the outsourcing request, the administrator node selects a distinctive task ID *Task_ID*, a key pair (*skey*, *pky*), and sends *pky* along with *Task_ID* to the user. Then, the user and the blockchain administrator node can negotiate their own session key *sskey* by exploiting Intel SGX technology. *sskey* should be kept from others. The stage corresponds to steps 1, 2, 3 and 4 in Fig. 3.
2) The user segments and uploads F to the blockchain administrator node. Optionally, concerning data privacy, users are allowed to choose a symmetric encryption algorithm and a secret key for encrypting their outsourcing file blocks. Then, the administrator node encapsulates the user's request and *Task_ID* to a task and then pushes it into the task queue. A certain number of tasks are organized as a task set. The blockchain administrator node allocates a thread pool for a task set, and those tasks within can be processed in parallel, which will be discussed later. The task dispatcher distributes a task according to the idle state of the system. In theory, this design can effectively accelerate the outsourcing process. However, its actual performance is also affected by the number of execution cores on the machine. The stage corresponds to step 5 in Fig. 3.
3) The blockchain administrator node retrieves the encrypted file blocks within a task and moves them to an enclave, which provides CPU hardware-level protection and memory encryption by isolating code and data from anyone. Then, the administrator node verifies and may also decrypt F with *sskey*, which is executed on SGX-capable Intel CPUs. With the verified F, the administrator node is able to perform user-specified operations, i.e., computing file authenticators for a PDP scheme. The execution status $status \in \{success, failure\}$ should be encrypted with the session key *sskey*. For simplicity, the encoded *status* is denoted as $task_c$. Similarly, other results, such as file authenticators, are also sealed as $task_o$. In the end, sChain encapsulates $status||task_c||task_o$ into a new task and pushes it into the receipt queue. The stage corresponds to step 6 in Fig. 3.
4) The blockchain administrator node retrieves a task sealed with $status||task_c||task_o$ from the receipt queue and checks the execution status. If the task fails, sChain rejects the outsourcing request and sends a failure message to the user as the response. Otherwise, $task_o$ is sent to the off-chain storage, and $task_c$ is sent to the user in response to his request. Optionally, the execution result can also be outsourced to blockchain for post audits. The stage corresponds to steps 7, 8 and 9 in Fig. 3.

For sChain, those burdensome tasks, such as computing file authenticators, are undertaken by a blockchain administrator node. However, for the heavy duty business, the increasing outsourcing files may overload the blockchain administrator node. This is common for the consortium blockchain. For example, in the copyright protection field, a user needs to store large files up to hundreds of gigabytes, such as videos and audios, and the blockchain administrator node needs to process them at a time. A feasible solution is to split the task and assign these subtasks to different computing nodes for handling in parallel. We propose another improved framework for the scenario with heavy computational burden, as demonstrated in Fig. 4. For the convenience of discussion, the aforementioned data outsourcing solution is marked as the single computing node framework (SCNF), while the improved solution is marked as the multiple computing node framework (MCNF). Compared with SCNF, there exist several differences in MCNF. First, in MCNF, a group of computing nodes is arranged into a worker pool, which is supervised and maintained by a blockchain administrator node. Each computing node retrieves and handles a task set at a time, and those tasks within can be executed in parallel. The corresponding results should be pushed into a receipt queue. Second, for MCNF, when a computing node powered by Intel SGX CPUs applies to join a worker pool, the node should attest its hardware and software configurations to the blockchain administrator node. If the attestation fails, the computing node's application will be rejected. Third, to facilitate the management of each worker pool in the MCNF, the administrator node requires each computing node to generate a private/public key pair (*skey*, *pky*). The private key *skey* is kept secretly in the cryptographic hardware of the computing node, while the public key *pky* is registered with blockchain in the form of a transaction. (*skey*, *pky*) can help build a secure channel between the computing

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

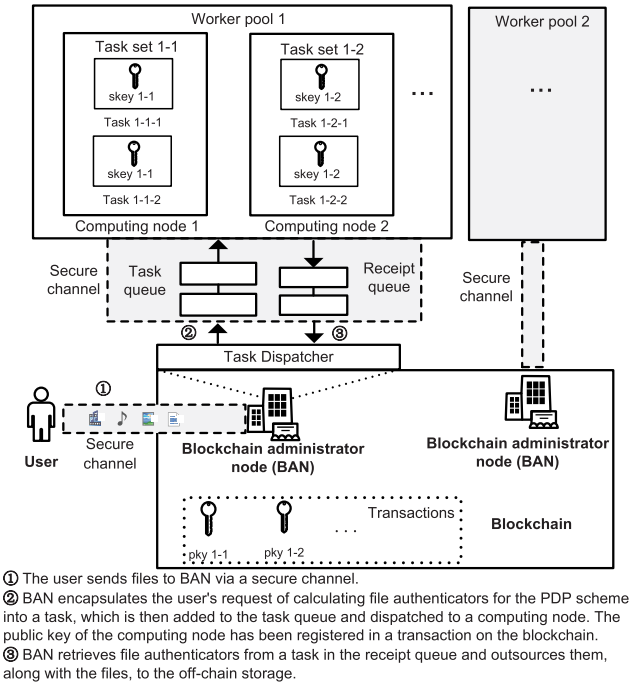


Fig. 4. Improved outsourcing framework.

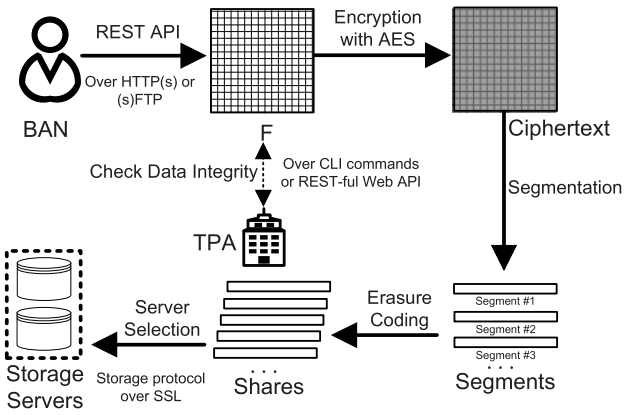


Fig. 5. Framework of integrating a PDP scheme into Tahoe-lafs.

node and the blockchain administrator node through the Diffie-Hellman key exchange protocol built in the Intel SGX technology. *pky* can be used to uniquely identify the computing node.

The blockchain administrator node is responsible for calculating file authenticators required by the proposed PDP scheme. Once it fails, the entire system will become unavailable. There are two countermeasures against the challenge. The first one is that we can consider introducing a faulty node detector to sChain. Once the administrator node is detected to be unavailable, we should select a new one from the blockchain network. The second one is that the user assumes the role of the blockchain administrator node, and needs to undertake the task of calculating file authenticators, which is feasible for applications with light outsourcing workload.

The excellent redundant backup feature of blockchain ensures on-chain data security. However, for these off-chain data, due to hardware or software failures, hacker attacks, etc., they may be broken and unavailable. To handle this challenge, we propose a new PDP scheme to verify the data integrity efficiently, which will be discussed in Section IV.

C. Discussion

We also need to consider how to integrate the off-chain storage to sChain. For this, we take Tahoe-lafs as an example to illustrate our solution. Tahoe-lafs is an open source distributed cloud storage system, and can distribute data segments to multiple servers. Even if some nodes fail, the entire system continues to function correctly. Tahoe-lafs comes with a tool called File Verifier to ensure data integrity. The tool will first download the ciphertext corresponding to the outsourcing data, and then leverages a hash function to perform integrity verification. Once the ourscourcing data are checked as corruption, Tahoe-lafs will leverage another tool File Repairer to regenerate the data content.

The new implementation which integrates the PDP scheme and Tahoe-lafs into sChain is demonstrated as Fig. 5. As shown in Fig. 1, when the blockchain management node outsources the file F and file authenticators to the off-chain storage, Tahoe-lafs encrypts and splits the data into fixed-size segments, from which we can calculate a set of shares by leveraging the erasure coding technology. Finally, a server selection algorithm is used to distribute these shares to the servers. The PDP scheme can retrieve the user data and the corresponding file authenticators, and then perform the data integrity verification through invoking CLI commands or REST-ful web API provided by Tahoe-lafs. Since the foregoing procedure does not revise the data content, it can run as a daemon in parallel with normal business, which will not reduce the system performance.

In sChain, users have the option to encrypt their data when outsourcing them. Tahoe-lafs provides a file management system, allowing data owners to access their files on the off-chain storage from a web platform. When the outsourced data are encrypted, the data receiver can locate the ciphertext data but not decrypt them. To access the plain text, they must negotiate with the data owner offline to obtain the decryption key. To make the process more efficient, it is promising to introduce a searchable encryption algorithm into sChain, allowing the data receiver to search the encrypted data with keywords. We will not discuss the searchable encryption scheme here, and readers can refer to the literature [41], [42] for more information.

IV. Overview

A. Algorithm Model

The proposed PDP scheme for sChain contains the following six algorithms: `Setup`, `DataProg`, `GenAuth`, `Challenge`, `Response` and `Verification`.

`Setup`: This is run by sChain to output the system public parameters $\{p, g\}$, the symmetric encryption function $Enc()$, and the secret key sk for $Enc()$. Note that $Enc()$ and its key sk

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

are optional. It depends on whether the user decides to protect data privacy.

`DataProg`: The algorithm is run by the user. Upon input of the outsourcing file F, it outputs the corresponding file tag FT and the encrypted file blocks $\bar{F}$, and then sends FT and $\bar{F}$ to the blockchain administrator node.

`GenAuth`: The algorithm is run by the blockchain administrator node. Upon input of FT and $\bar{F}$, it outputs the authenticator set δ, the public key K and the value set A. Then, the administrator node should outsource δ and $\bar{F}$ to the off-chain storage, FT, A and K to the blockchain.

`Challenge`: This is run by the TPA. Upon input of the number of challenged file blocks l, it outputs the challenge set Q, which should be sent to the off-chain storage.

`Response`: The algorithm is run by the off-chain storage. Upon input of the challenge set Q, the off-chain storage calculates the corresponding proof $proof = \{U, \sigma\}$, which is sent to the TPA for verification.

`Verification`: This is run by the TPA. Upon input of $proof = \{U, \sigma\}$ received from the off-chain storage, it outputs the verification result.

B. Security Model

When the data are outsourced to the off-chain storage, sChain should guarantee that the data cannot be tampered with or lost. The underlying PDP scheme of sChain can check the data integrity with low communication costs. We revise the security model proposed by [3], which focuses on auditing soundness, and apply it to our proposed PDP scheme. The PDP scheme is sound if the verifier can be convinced that any prover's declaration of storing an intact file is true. For the security model, we define a game between an adversary $\mathcal{A}$ (i.e., dishonest off-chain storage) and a challenger $\mathcal{C}$ (i.e., TPA). There is an extractor $Extr(sk, FT, \mathcal{P}')$, which takes the secret key sk, the file tag FT and some extra descriptions $\mathcal{P}'$ as inputs and outputs the file F.

The interaction details between adversary $\mathcal{A}$ and challenger $\mathcal{C}$ are as follows.

- $\mathcal{C}$ runs the `Setup` algorithm to obtain system public parameters and the secret key sk. These parameters should be forwarded to $\mathcal{A}$.
- $\mathcal{A}$ interacts with $\mathcal{C}$ by querying to the `DataProg` oracle with F as input. The challenger $\mathcal{C}$ should respond to the adversary by running the `DataProg` algorithm.
- $\mathcal{A}$ interacts with $\mathcal{C}$ by running the `Challenge` algorithm and the `Response` algorithm with F as query input. $\mathcal{C}$ as the verifier should run the `Verification` algorithm to check the response of the prover $\mathcal{A}$.
- Finally, $\mathcal{A}$ can output FT and a subalgorithm $\mathcal{P}'$ through those previous queries.

We say $\mathcal{P}'$ is ε-admissible if more than $1/\varepsilon$ of the prover's responses to the verifier's challenges can pass the `Verification` algorithm. To measure the status of data integrity for the PDP scheme, we introduce the following definition.

Definition 1: A PDP scheme is ε-sound if there exists an extractor $Extr()$ such that for any $\mathcal{A}$ and $\mathcal{C}$ that play the challenge-response game and output the ε-admissible $\mathcal{P}'$ for F, during which $Extr()$ can extract the original file F except with negligible probability.

Intel SGX is an emerging technology with some associated security vulnerabilities, such as susceptibility to side-channel attacks. To address this, Intel Corporation and researchers have developed optimization solutions for real-world applications [43]. Although Intel Core processors of the 11th and 12th generations no longer support SGX in 2021, Intel Xeon processors are still employing the technology to provide services for enterprise applications. For example, Alibaba Cloud is offering users security-enhanced instances, like g7t, c7t, and r7t, which are based on Intel SGX technology, to create a secure and confidential computing environment [43]. In light of that, we assume that the specified procedure of an SGX application will be correctly and safely executed.

sChain does not specify the underlying blockchain implementation or the corresponding consensus protocol. In fact, sChain can be deployed atop any consortium blockchain, such as Hyperledger Fabric and Quorum. The corresponding PDP scheme should be adapted to the decentralized framework of blockchain. We assume that the chosen blockchain implementation can perform the prescribed computation correctly, and the on-chain data should be immutable. To improve efficiency, sChain can choose an effective consensus protocol, such as Practical Byzantine Fault Tolerance (PBFT), to quickly construct blocks. PBFT does not require any confirmations and is more efficient than most consensus protocols, such as Proof of Stake (PoS) [44].

C. Design Goals

Here, we summarize the design goals of the PDP scheme for sChain. Briefly, the PDP scheme aims to help build an efficient decentralized method to ensure the integrity of off-chain data. Here data integrity guarantees that the off-chain storage can faithfully store the outsourced data.

1) Decentralized Key Management: To better adapt to the decentralized architecture of sChain, the new PDP scheme should not store any secret keys and thus does not need any centralized entity to maintain these keys, which can eliminate a single point of failure introduced by such a design. sChain allows users to select an encryption algorithm along with a secret key to protect their data, but it is optional. Even if the user loses the secret key, sChain can still perform the data auditing normally.

2) Correctness: When the off-chain data are intact, if the entities involved in sChain are honest and obey the specified procedures, the proof generated by the off-chain storage can pass the TPA's verification.

3) Auditing Soundness: If the off-chain data are broken, the corresponding storage system cannot generate a correct response to pass the TPA's verification.

4) Public Auditing: A TPA who does not hold any exclusive private keys can verify the integrity of off-chain data, which can save sChain considerable hassle in maintaining user keys.

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

D. Preliminaries

1) Computational Diffie-Hellman (CDH) Problem: For $a, b \in \mathbb{Z}_p^*$, given g, $y = g^a$, g^b as input, output y^b. The CDH assumption holds if it is computationally infeasible to solve the CDH problem.

2) Discrete Logarithm (DL) Problem: For $a \in \mathbb{Z}_p^*$, given g, g^a as input, output a. The DL assumption holds if it is computationally infeasible to solve the DL problem.

V. Concrete Construction

A. Design Rationale

It is promising to leverage off-chain storage to help increase the storage capacity of blockchain. However, one major challenge of designing such a scheme is how to ensure the security of the off-chain data. It urges us to revise a new strategy of detecting data corruption quickly at low cost. For this, we utilize Intel SGX technology and a proposed PDP scheme to verify the off-chain data integrity. We select a participant with Intel SGX enabled in the blockchain network as the administrator node, which should authenticate its hardware and software configurations to users. The blockchain administrator node, which serves as a trusted entity, is responsible for calculating file authenticators and outsourcing them to the off-chain storage, which can accelerate the proposed PDP scheme through parallelization. For an Intel SGX-enabled application running on the blockchain administrator node, because the enclave memory is limited in size, the node cannot handle a large file at a time. It is promising that the file should be segmented and assigned to different entities for processing in parallel. However, due to hardware or software failures, the administrator node may crash and become unavailable. Therefore, the blockchain administrator node should not retain any data after the `GenAuth` algorithm terminates. When executing the `Challenge` algorithm, because the proposed PDP scheme supports public auditing, the TPA does not need to hold the data owner's secret key. In theory, the blockchain administrator node or any user can take the role of TPA and verify the off-chain data.

Another challenge is how to design an efficient decentralized paradigm for the PDP scheme so that it can be incorporated into blockchain with low adaptation costs. The decentralization feature of blockchain alludes to the transfer of control from a centralized authority to a dispersed network. Our proposed PDP scheme should eliminate any centralized key distribution center because it may incur a single point of failure. When users outsource their files to the off-chain storage, these computation-intensive operations of the PDP scheme can be undertaken by the blockchain administrator node, which is in charge of a set of computing nodes. One advantage is that the execution efficiency of sChain no longer depends on the computing power of the user device, so the new proposed system can be adapted to a wide range of businesses. Compared with the public blockchain, the consortium blockchain can store data faster and more efficiently and has been applied to a wider range of businesses. sChain is promising for the consortium blockchain, such as Hyperledger Fabric. This is because sChain can improve its storage capacity efficiently and protect the data security with low costs.

TABLE I
Notations

Symbols	Descriptions
p	safe prime number
$\mathbb{Z}_p^*$	cyclic group module the prime number p
g	generator of $\mathbb{Z}_p^*$
F	file to be stored
$\overline{F}$	encrypted file
Fs	size of F
r	percentage of F to be audited
$Enc\,()$	symmetric encryption function
sk	secret key for $Enc\,()$
n	number of file blocks
s	number of sectors for each file block
c	encrypted file block
FT	file tag
$H_1\,()$	pseudorandom hash function
δ	file authenticators
Q	challenge set
l	number of random values in the set Q
$proof$	proof message from the off-chain storage
(k_{prf}, K)	private/public key pair
$a_1, ..., a_j, ..., a_s$	random numbers from $\mathbb{Z}_p^*$
$A_1, ..., A_s$	values calculated by $g^{a_1}, ..., g^{a_s}$
A	set of $\{A_j\}_{1\le j\le s}$

B. Concrete Construction

In Table I, we list some common notations to help readers better understand the proposed PDP scheme. The detailed description of the scheme is as follows.

`Setup`: This algorithm is run to output the system public parameters.

1) sChain selects a large prime number p. g is the generator of $\mathbb{Z}_p^*$.
2) $Enc\,()$ is a standard symmetric encryption function such as AES. The secret key of $Enc\,()$ is denoted as sk.
3) The sChain chooses a pseudorandom function, which is $H_1 : \{0,1\}^p \rightarrow \mathbb{Z}_p^*$

`DataProg`: For the file F to be uploaded to the off-chain storage, the user needs to encrypt and split F into n blocks, and then sends them to the blockchain administrator node.

1) The user randomly selects a secret key $sk \in \mathbb{Z}_p^*$ for the function $Enc\,()$, and then encrypts F to obtain the encrypted file $\bar{F} \leftarrow Enc_{sk}(F)$.
2) Split the file $\bar{F}$ into n blocks, and each block has s sectors. For simplicity, we denote $\bar{F}$ as $\{c_{ij}\}_{1\le i\le n, 1\le j\le s}$.
3) Obtain the file tag $FT = n \parallel s \parallel Fs$. Fs denotes the size of F.
4) The user runs a secure remote attestation protocol along with the blockchain administrator node, which is powered by Intel SGX, so the administrator node can authenticate its hardware and software configurations to the user. Afterwards, the built-in AES-GCM algorithm

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

of the SGX technology can help create a secure channel between the user and the administrator node.

5) The user sends the file tag FT and $\bar{F}$ to the blockchain administrator node through the secure channel.

`GenAuth`: With FT and $\bar{F}$, the blockchain administrator node produces the authenticator set for $\bar{F}$.

1) Parse the file tag FT to obtain n, s and Fs
2) Choose s random numbers $\{a_j \in \mathbb{Z}_p\}_{1\le j\le s}$ and a private key $k_{prf} \in \mathbb{Z}_p$.
3) Calculate the public key $K = g^{k_{prf}}$.
4) For $1 \le j \le s$, calculate $A_j = g^{a_j}$. For simplicity, we denote $A = \{A_j\}_{1\le j\le s}$.
5) For $1 \le i \le n$, calculate $\delta_i = H_1(i)\, k_{prf} + \sum_{j=1}^{s} (a_j c_{ij})$. We denote the authenticator set δ as $\{\delta_i\}_{1\le i\le n}$.
6) Outsource δ and $\bar{F}$ to the off-chain storage, $\{A, K\}$ and FT to the on-chain storage.
7) Delete $\{a_j \in \mathbb{Z}_p\}_{1\le j\le s}$ and $k_{prf} \in \mathbb{Z}_p$ from the local storage.

`Challenge`: The TPA can decide the percentage of data volume to be audited according to the computing power or other preset conditions. For example, when the system is idle, the TPA can increase the percentage r, $0 \le r \le 1$, otherwise, r should be decreased. The TPA retrieves FT from the on-chain storage, and then chooses a challenge set of l different random elements (i, v_i), such that $i \in [1, n]$, $\|v_i\| < \|p\|$, $l = \lceil Fs \times r \rceil$. Denote the challenge set as $Q = \{(i_1, v_{i_1}), \ldots, (i_l, v_{i_l})\}$. The TPA sends Q to the off-chain storage afterwards.

`Response`: With the challenge set Q, the off-chain storage calculates a corresponding proof and demonstrates the data in its possession remain intact.

1) For $1 \le j \le s$, calculate $u_j = \sum_{(i,v_i)\in Q} (v_i c_{ij})$. We denote $U = \{u_1, u_2, \ldots, u_s\}$.
2) Calculate $\sigma = \sum_{(i,v_i)\in Q} (v_i \delta_i)$.
3) Send $proof = \{U, \sigma\}$ to the TPA for verification.

`Verification`: The TPA checks the correctness of *proof* received from the off-chain storage.

1) Parse $proof = \{U, \sigma\}$, and fetch the corresponding elements $\{A, K\}$ from the on-chain storage.
2) Check whether the equation $g^{\sigma} = K^{\sum_{(i,v_i)\in Q} (v_i H_1(i))} \prod_{j=1}^{s} {A_j}^{u_j}$ holds. If it holds, sChain returns "true"; otherwise returns "false".
3) Store the result on blockchain for auditing.

C. Correctness Analysis

For the challenge set $Q = \{(i_1, v_{i_1}), \ldots, (i_l, v_{i_l})\}$ and the valid corresponding *proof*, the equation $g^{\sigma} = K^{\sum_{(i,v_i)\in Q} (v_i H_1(i))} \prod_{j=1}^{s} {A_j}^{u_j}$ in the `Verification` algorithm holds.

Proof:

$$
\begin{aligned}
g^{\sigma} &= g^{\sum_{(i,v_i)\in Q} (v_i \delta_i)} \\
&= g^{\sum_{(i,v_i)\in Q} \left(\left(H_1(i) k_{prf} + \sum_{j=1}^{s} (a_j c_{ij}) \right) v_i \right)} \\
&= g^{k_{prf} \sum_{(i,v_i)\in Q} (v_i H_1(i))} \; g^{\sum_{(i,v_i)\in Q} v_i \left(\sum_{j=1}^{s} (a_j c_{ij}) \right)} \\
&= K^{\sum_{(i,v_i)\in Q} (v_i H_1(i))} \; g^{\sum_{j=1}^{s} \left(a_j \left(\sum_{(i,v_i)\in Q} (v_i c_{ij}) \right) \right)} \\
&= K^{\sum_{(i,v_i)\in Q} (v_i H_1(i))} \; g^{\sum_{j=1}^{s} (a_j u_j)} \\
&= K^{\sum_{(i,v_i)\in Q} (v_i H_1(i))} \prod_{j=1}^{s} g^{(a_j u_j)} \\
&= K^{\sum_{(i,v_i)\in Q} (v_i H_1(i))} \prod_{j=1}^{s} {A_j}^{u_j}
\end{aligned}
$$

VI. Security Analysis

Theorem 1 (Auditing soundness): For the proposed PDP scheme, if the off-chain data are intact, the response from the off-chain storage can pass the TPA's verification.

Proof: We construct a knowledge extractor *Extr*() to complete the proof [11]. If the off-chain data are not intact, but the response can pass the TPA's verification, then we can utilize *Extr*() to extract the intact challenged file blocks through performing the interaction described in Section IV-B.

Game 0: The challenger $\mathcal{C}$ runs the `Setup` algorithm to obtain the system public parameters, which will be sent to the adversary $\mathcal{A}$. Then, $\mathcal{A}$ picks out a set of file blocks $\{c_{ij}\}_{1\le i\le n, 1\le j\le s}$, and submits them to the challenger $\mathcal{C}$. Upon receiving them, $\mathcal{C}$ performs the `DataProg` algorithm to obtain the file tag FT and the encrypted file blocks, and runs the `GenAuth` algorithm to fetch the corresponding authenticator sets δ, A and K. The challenger should send FT, δ, A, K and the encrypted file blocks to $\mathcal{A}$. When to verify the data integrity, the challenger $\mathcal{C}$ sends a challenge set Q to the adversary. On receiving Q, the adversary calculates the message $proof$ as the response. The challenger checks $proof$ afterwards. If $proof$ can pass through the verification, the adversary wins.

Game 1: Game 1 is almost the same as Game 0, except for one difference. That is, in the process of performing the `DataProg` step, the challenger records all the file tags and stores them in a list L. If the adversary succeeds in constructing a file tag FT, but is not within L, the challenger aborts the game and the adversary wins.

Analysis: If the challenger can abort the game with a non-negligible probability, it means that the adversary can successfully forge the file tag FT, which is storing on blockchain. It contradicts that the on-chain data are immutable as described in Section IV-B. So the block count n, the number of sectors s and the file size Fs in the interactions between $\mathcal{C}$ and $\mathcal{A}$ are all correct.

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

Game 2: Game 2 is the same as Game 1, except for one difference. That is, the challenger records all the responses to the queries from the adversary. When issuing a challenge, the challenger observes all the responses from the adversary. If σ is not equal to $\sum_{(i,v_i)\in Q}(v_i\delta_i)$, the challenger aborts the game and the adversary wins.

Analysis: Assuming that the correct response is $proof = \{U, \sigma\}$, since it can pass the challenger's verification, the following equation should hold.

$$g^{\sigma} = K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))} \prod_{j=1}^{s} A_j{}^{u_j} \tag{1}$$

Assuming that the adversary successfully forges a response $proof' = \{U', \sigma'\}$, which can be verified correct, the following equation holds.

$$g^{\sigma'} = K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))} \prod_{j=1}^{s} A_j{}^{u_j'} \tag{2}$$

Obviously, $U' \neq U$, otherwise $\sigma' = \sigma$. Dividing the equation (2) by the equation (1), we have

$$g^{\sigma'-\sigma} = \prod_{j=1}^{s} A_j{}^{u_j'-u_j} = g^{\sum_{j=1}^{s}\left(a_j\left(u_j'-u_j\right)\right)} \tag{3}$$

For the forged $proof'$ from the adversary, based on the CDH assumption, we prove that the adversary can not make the challenger abort the game with a non-negligible probability.

Given g', g'^{x}, $w \in \mathbb{Z}_p^*$, the challenger works for calculating w^x.

The challenger sets $g = g'w$, and generates the encrypted file blocks $\{c_{ij}\}_{1\le i\le n, 1\le j\le s}$ and $FT = n \parallel s \parallel Fs$. The adversary does not know the plain text corresponding to those encrypted file blocks, otherwise it contradicts that the encryption algorithm $Enc()$ is secure.

For each i in the challenge, the challenger selects a random element $v_i \in \mathbb{Z}_p^*$, so we can get $u_j = \sum_{(i,v_i)\in Q}(v_i c_{ij}), 1 \le j \le s$. Moreover, the challenger programs the random oracle at i as $H_1(i) = \left(x - \sum_{j=1}^{s}(a_j c_{ij})\right)/k_{prf}$, so we can get

$$\begin{aligned}\sigma &= \sum_{(i,v_i)\in Q}(v_i\delta_i)\\ &= \sum_{(i,v_i)\in Q}\left(v_i\left(H_1(i)\,k_{prf} + \sum_{j=1}^{s}(a_j c_{ij})\right)\right)\\ &= \sum_{(i,v_i)\in Q}(v_i x)\end{aligned} \tag{4}$$

Then we can know that

$$g^{\sigma'-\sigma} = g^{\sum_{(i,v_i)\in Q}(v_i'x) - \sum_{(i,v_i)\in Q}(v_i x)} \tag{5}$$

From equations (3) and (5), we can obtain that

$$\begin{aligned}g^{\sigma'-\sigma}/g^{\sum_{j=1}^{s}\left(a_j\left(u_j'-u_j\right)\right)} &= g^{\sigma'-\sigma-\sum_{j=1}^{s}\left(a_j\left(u_j'-u_j\right)\right)}\\ &= g^{x\sum_{(i,v_i)\in Q}(v_i'-v_i)-\sum_{j=1}^{s}\left(a_j\sum_{(i,v_i)\in Q}((v_i'-v_i)c_{ij})\right)}\\ &= 1\end{aligned}$$

Since $g = g'w$, we can calculate

$$\left(g'w\right)^{x\sum_{(i,v_i)\in Q}(v_i'-v_i)-\sum_{j=1}^{s}\left(a_j\sum_{(i,v_i)\in Q}((v_i'-v_i)c_{ij})\right)} = 1$$

From the above equation, we can obtain that $w^x = \left(\left(g'w\right)^{\sum_{j=1}^{s}\left(a_j\sum_{(i,v_i)\in Q}((v_i'-v_i)c_{ij})\right)}\right)^{\sum_{(i,v_i)\in Q}(v_i'-v_i)^{-1}} / \left(g'\right)^{x}$

The probability that we cannot solve the CDH hard problem is the same as the probability of $\sum_{(i,v_i)\in Q}\left(v_i' - v_i\right) = 0$. Since the challenged blocks number is 1, the probability is $(1/p)$, which is negligible. It means that the challenger can solve the CDH hard problem if the difference between the adversary's probabilities of success in Game 1 and Game 2 is non-negligible.

Game 3: Game 3 is the same as Game 2, with one difference. The challenger still needs to observe and store all the responses to the queries from the adversary. The difference between Game 3 and Game 2 is as follows. If any u_j is not equal to $\sum_{(i,v_i)\in Q}\left(v_i c_{ij}\right)$, the challenger declares failure and aborts the game.

Analysis: Assuming the correct response is $proof = \{U, \sigma\}$, and the forged one is $proof' = \{U', \sigma'\}$. Since both $proof$ and $proof'$ can pass the challenger's verification, the following equations should hold.

$$\begin{aligned}g^{\sigma} &= K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))}\prod_{j=1}^{s} A_j{}^{u_j}\\ &= K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))}\, g^{\sum_{j=1}^{s}(a_j u_j)}\end{aligned} \tag{6}$$

$$\begin{aligned}g^{\sigma'} &= K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))}\prod_{j=1}^{s} A_j{}^{u_j'}\\ &= K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))}\, g^{\sum_{j=1}^{s}(a_j u_j')}\end{aligned} \tag{7}$$

Game 2 has proved the equation $\sigma' = \sigma$ holds, and we will construct a simulator to solve the DL hard problem.

Since

$$\begin{aligned}g^{\sigma} &= K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))}\, g^{\sum_{j=1}^{s}(a_j u_j)}\\ &= g^{\sigma'}\\ &= K^{\sum_{(i,v_i)\in Q}(v_i H_1(i))}\, g^{\sum_{j=1}^{s}\left(a_j u_j'\right)},\end{aligned}$$

we have $g^{\sum_{j=1}^{s}\left(a_j u_j'\right) - \sum_{j=1}^{s}(a_j u_j)} = 1$.

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

TABLE II
SYSTEM PARAMETERS

Descriptions	Parameters settings	Descriptions	Parameters settings
Blockchain platform	Quorum 21.4.0	Off-chain storage platform	Tahoe-lafs 1.14.0
# of consensus nodes	3	# of storage servers	3
Consensus protocol	RAFT	# of shares when applying erasure-coding	10
Language for implementing smart contracts	Solidity	Segment size for off-chain storage platform	128 KB
SGX version	Intel 2.14 SGX SDK		

Set $\Delta = \sum_{j=1}^{s} \left(a_j u'_j\right) - \sum_{j=1}^{s} \left(a_j u_j\right)$, and we can get

$$g^{\Delta} = 1. \tag{8}$$

Given w, $g' \in \mathbb{Z}_p^*$, the challenger tries to compute a value $x \in \mathbb{Z}_p^*$ which satisfies $w = g'^x$.

The challenger selects a random value $a \in \mathbb{Z}_p^*$, and sets $g = g'w^a$. From the equation (8), we have

$$1 = g^{\Delta} = g'^{\Delta} w^{a\Delta}$$

Since $\Delta \neq 0$, we can obtain

$$w = \left(g'\right)^{(-\Delta)/(a\Delta)} = \left(g'\right)^{(-1)/(a)}$$

The probability of $a = 0$ is $1/p$, which is negligible. Therefore, we can solve the DL problem with a probability $1 - (1/p)$, which contradicts the assumption that the DL problem is hard. It means that the challenger can solve the DL hard problem if the difference between the adversary's probabilities of success in Game 2 and Game 3 is non-negligible.

Therefore, the differences between the above games can be ignored.

Finally, we construct *Extr*() to extract l challenged file blocks c_i, $i \in Q$, $l = \|Q\|$ as the literature [28]. We execute l different challenges on the same data blocks c_i. This results in the knowledge extractor obtaining l independent linear equations. By solving these equations, the knowledge extractor is able to compute and extract c_i. This means that if the cloud can pass the TPA's verification, the user's data are guaranteed to remain intact.

From the previous discussion, if the off-chain storage can pass the TPA's verification, its storing data should be intact.

Theorem 2 (Detectability): In the proposed PDP scheme, for an outsourcing file with n blocks, when m blocks are broken and TPA tries to challenge l blocks, the probability that the data corruption is detected is at least $1 - ((n-m)/n)^l$.

Proof: The proof is similar to that in [28], and we omit the detailed analysis due to space limitations.

VII. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed system. The evaluation was conducted with Alibaba Cloud Elastic Compute Service (ECS), and the instance family was ecs.c7t. The detailed configurations were as follows. OS: Alibaba Cloud Linux 2.1903 LTS 64 bits; CPU: Intel(R) Xeon(R) Platinum 8369B CPU @ 2.70 GHz; RAM: 8 GB.

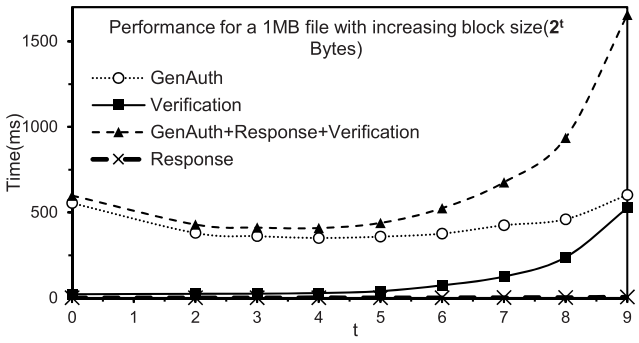


Fig. 6. Optimal block size.

In our setting, we selected Quorum as the underlying permissioned blockchain platform and a 1024-bit large prime number p. We do not consider the network delay so as to focus on the computational complexity. Table II lists all the system parameters used in the experiment.

A. Evaluation of the Proposed PDP Scheme

First, we try to find the optimal file block size s. Since s affects the running time of the three algorithms `GenAuth`, `Verification` and `Response` in the proposed PDP scheme, we test the time consumption to find the optimal block size. We fix the test file as 1 MB and increase the file block size to test the execution time of the preceding three algorithms. According to theoretical analysis, since the file size is fixed, as the block size increases, the number of blocks will decrease. For `GenAuth`, it will take more time to compute A_j and less time to compute δ_i. When the TPA performs the `Challenge` algorithm, we fix the number of challenged file blocks as 30. As the block size increases, computing u_j in the `Response` algorithm will cost more time. Likewise, the `Verification` algorithm will also be more time-consuming. As illustrated in Fig. 6, we test the time consumption of the `GenAuth`, `Verification` and `Response` algorithms. We can see that the optimal block size is 16 bytes, and the corresponding sum of the running times of the three algorithms is 409 ms.

Second, we compare the execution time of each algorithm for our new PDP scheme with the public PDP scheme proposed by Zhang et al. in 2020 [28]. Similarly, we fix the file size as 1 MB, the block size as 16 bytes, and the challenged block number as 30. The evaluation results are shown in Fig. 7. The private key generation algorithm in Zhang et al.'s scheme corresponds to `DataProg` in Fig. 7. For our proposed scheme, `GenAuth` takes the longest time, accounting for

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

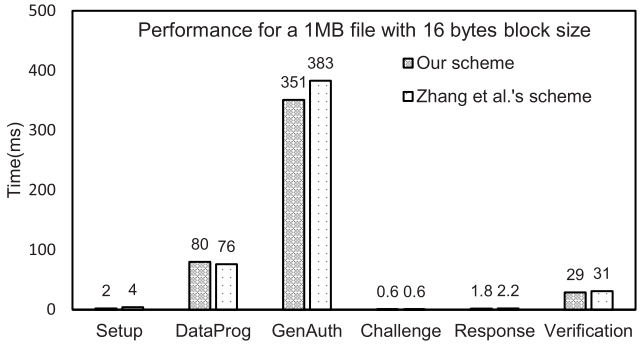


Fig. 7. Execution time of each algorithm.

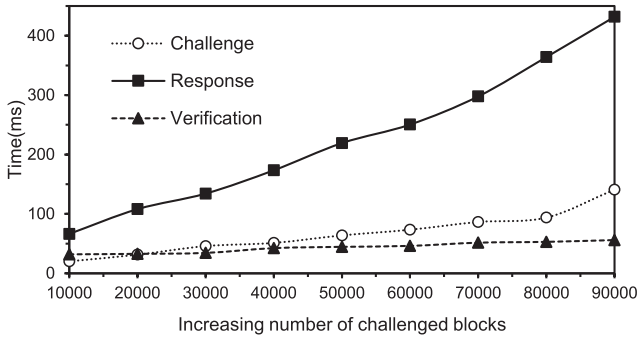


Fig. 8. Performance of three frequently executed algorithms for the proposed scheme.

approximately 75% of the total running time of these six algorithms in the proposed PDP scheme. Next is `DataProg`, which takes approximately 17% of the total running time. Fortunately, the above two algorithms only need to be performed once when outsourcing a file block. In contrast, the `Challenge`, `Response` and `Verification` algorithms need to be frequently executed in the life cycle of a file block. Zhang et al.' scheme presents similar results, but the `GenAuth` and `Verification` algorithms consume more time. That is because these two algorithms comprise expensive bilinear pairing operations. For the most time-consuming `GenAuth` in our PDP scheme, we accelerate the algorithm by leveraging Intel SGX technology and parallelization.

Third, we test the time cost of the `Challenge`, `Response` and `Verification` algorithms with the increasing volume of auditing data. We fix the block count as 90000 and the block size as 16 bytes. As shown in Fig. 8, the time consumption of each algorithm is almost linear with the challenged file blocks. Specifically, the `Response` algorithm takes more time than the sum of the execution time of the `Challenge` and `Verification` algorithms. For the `Response` algorithm, we can speed it up with a scalable parallelization strategy. Users can disperse their outsourcing file blocks to different off-chain storage devices. When performing the `Response` algorithm, each of the corresponding off-chain devices can calculate u_j and σ with its own stored data individually, which are then merged by the TPA to obtain the final results.

B. Evaluation of sChain

Currently, there are two primary approaches for auditing off-chain data integrity as outlined in Section II. The first one is based on hash functions, wherein the auditor must obtain the audited data, compute their hash values, and then compare them to the stored values [4], [6]. The second one utilizes Merkle trees, wherein the auditor should obtain the audited data, and then compute and compare the hash value of the root node [9], [10]. For comparison, let n denote the number of file blocks to be audited. Table III compares sChain to the above two approaches in terms of storage cost, bandwidth cost, and computational complexity of verifying data integrity. Results indicate that sChain outperforms the above two approaches in terms of bandwidth cost and time complexity when verifying data integrity.

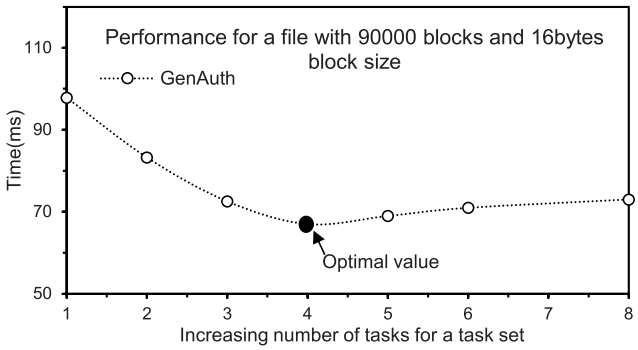


Fig. 9. Optimal task counts in a task set for SCNF.

Since the `GenAuth` algorithm is time-consuming, sChain will utilize Intel SGX technology for acceleration. We will attempt to find the optimal task counts for a task set in the proposed SCNF. In SCNF, a task set is assigned to a computing node, and those tasks within can be executed in parallel on the machine by allocating a thread for each task. To reduce the overhead of maintaining these threads, the blockchain administrator node will allocate a thread pool for a task set. We fix the number of file blocks as 90,000 and the block size as 16 bytes. Experimental results are shown in Fig. 9. As the number of tasks in the task set increases, the execution time of the `GenAuth` algorithm decreases first and then increases. When the task count is 4, we obtain the optimal execution time, which is 67 ms. The number of threads of our CPU is 4, which means that our machine can process 4 tasks in parallel. Therefore, to reach the optimal performance, we should fix task counts within a task set as the number of CPU threads.

Then, we try to find the optimal number of task sets for a worker pool as the tradeoff between performance and communication costs in MCNF. For the convenience of comparison, we also take the `GenAuth` algorithm as an example. In MCNF, a worker pool is undertaken by a set of computing nodes, and each node will be assigned with a task set. A blockchain administrator node is in charge of a worker pool. According to the theoretical analysis, with the increase in computing nodes in the worker pool, the degree of parallelism (DOP) is increased, while communication costs are higher. We conduct the following experiment to illustrate the tradeoff. First, the number of file blocks is fixed as 240000, and the block size is fixed as 16 bytes. We set the task count in a task set as 4. Experimental results are shown in Fig. 10. When the number of task sets in a worker pool

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

TABLE III
COMPARISON OF STORAGE COST, BANDWIDTH COST, AND COMPUTATIONAL COMPLEXITY

Approaches	Storage cost	Bandwidth cost	Computational complexity of verifying data integrity
Hash-based approaches	$O(n)$	$O(n)$	$O(n)$
Merkle tree based approaches	$O(1)$	$O(n)$	$O(n)$
sChain	$O(n)$	$O(1)$	$O(1)$

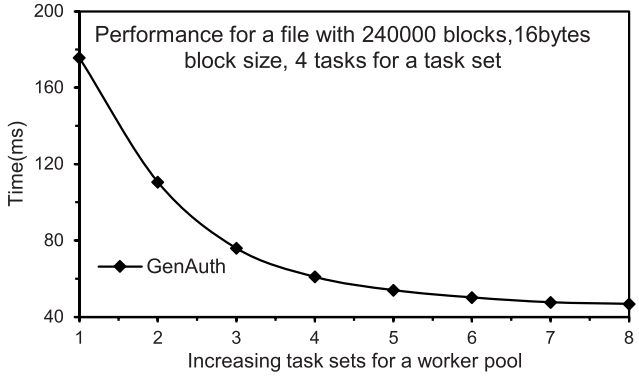


Fig. 10. Optimal number of task sets in a worker pool for MCNF.

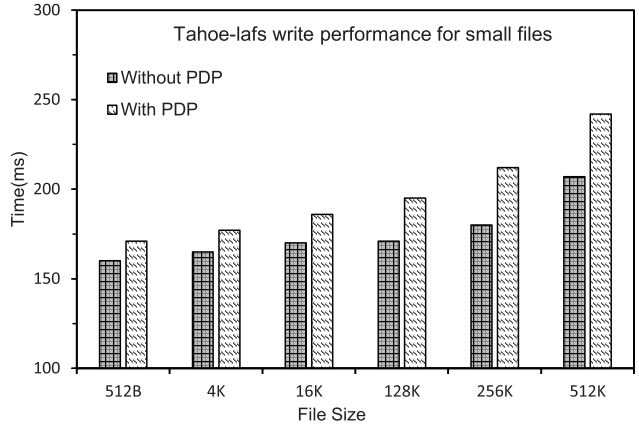


Fig. 11. Write performance for small files on Tahoe-lafs.

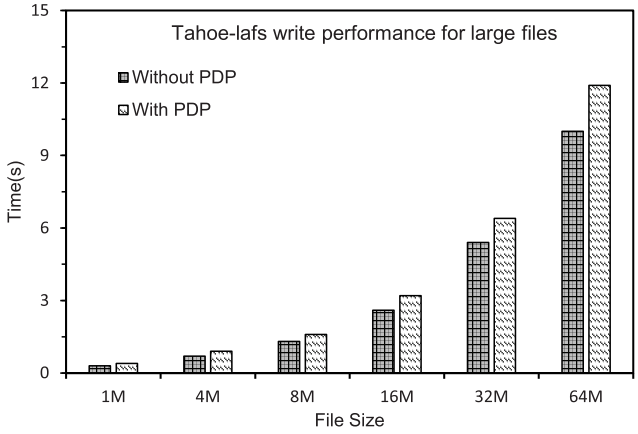


Fig. 12. Write performance for large files on Tahoe-lafs.

TABLE IV
READ/WRITE/CHECK PERFORMANCE OF TAHOE-LAFS

Operation	File Size	Time	Extra storage overhead
$Write$[1]	512B	160ms	–
	4K	165ms	–
	1M	349ms	–
$Read$[1]	512B	10ms	–
	4K	13ms	–
	1M	34ms	–
$Check$[1]	512B	32ms	–
	4K	39ms	–
	1M	76ms	–
$Write^{+2}$	512B	171ms	6.5KB
	4K	177ms	34.5KB
	1M	402ms	8MB
$Read^{+2}$	512B	11ms	0
	4K	13ms	0
	1M	36ms	0
$Check^{+2}$	512B	10ms	0
	4K	12ms	0
	1M	33ms	0

[1] Read/Write/Check performance for Tahoe-lafs with data auditing disabled.
[2] Read/Write/Check performance when applying the proposed PDP scheme into Tahoe-lafs.

increases from 1 to 8, the execution time of the `GenAuth` algorithm decreases at first and then remains constant. It can be concluded that when the blockchain administrator node handles a heavy workload, we should allocate more task sets for the worker pool. Conversely, we should reduce the number of task sets to save computing resources. For the strategy of how to dynamically allocate the number of task sets, there have been many related studies [45], [46]. We will not discuss it here because it is beyond the research scope of the paper.

Afterwards, we evaluate the auditing cost of Tahoe-lafs using our proposed PDP scheme. For this, we select the SCNF model, set the task count to 4, and the segment size to 16 bytes. When performing the data verification, Tahoe-lafs enabled with the proposed PDP scheme will challenge one-third of the stored data. Table. IV provides the read/write/check execution time, and the storage cost for processing files with sizes of 512B, 4K and 1M respectively. As displayed in Table IV, the PDP scheme decreases the write performance, but has a minor effect on retrieving these storing files. For example, for a file with 512 bytes, it takes about 160 ms to upload the file to Tahoe-lafs with data auditing disabled, and 10 ms to download the file. Then, when applying the PDP scheme to Tahoe-lafs, it takes about 171 ms to complete the uploading task, and almost the same time to retrieve the file. The PDP scheme does not only decrease the writing speed, but also significantly increase the storage consuming. For example, for a 1M file, the PDP scheme needs to store extra 8 MB of data, which include A, K, FT, and δ. Although the proposed PDP scheme increases the system writing overhead and storage cost, it can still improve the performance of data verification, and be effectively adapted to many applications such as verifying the system logs, due to the fact that the writing operation only needs to be performed once, while read and check operations should be invoked frequently.

Finally, we evaluate the write performance for different file sizes before and after applying the PDP scheme

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

into Tahoe-lafs. When uploading small files to Tahoe-lafs, the data writing time are shown as Fig. 11. As displayed in the figure, with the file size increasing ($\leq$128 KB), the writing speed for Tahoe-lafs enabled with data auditing decreases significantly, while the writing time for the native Tahoe-lafs system remains almost unchanged. This is because the default fragment size of Tahoe-lafs is 128K, and small files will be filled into fixed-size blocks of the same length before performing subsequent upload operations, so the time consumption for those small file are almost the same. The file writing time for outsourcing large files is shown as Fig. 12. We can see that the execution time of Tahoe-lafs increases significantly with the file size whether or not the PDP scheme is enabled. This is because for large files, the corresponding number of segmented file blocks is linearly related to the file size, and the more file blocks are uploaded, the more time-consuming it will be.

VIII. Conclusion

In this paper, we propose sChain, which can effectively improve the storage capacity of blockchain without revising its implementation. sChain leverages Intel SGX technology and parallel processing to construct a trusted computing environment and accelerate the process of data outsourcing. In addition, we propose a new PDP scheme to verify the off-chain data integrity, which can detect data corruption in time. The new PDP scheme does not need to hold any keys and allows any entity to check the data integrity, which can be well adapted to blockchain. We also provide security analyses and launch related evaluations. In the future, we will deploy sChain in an actual business to test the performance and further optimize its efficiency.

References

[1] S. Nakamoto. (2008). *Bitcoin: A Peer-To-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] K. E. Agbezoutsi, P. Urien, and T. M. Dandjinou, "Mobile money traceability and federation using blockchain services," *Ann. Telecommun.*, vol. 76, nos. 3–4, pp. 223–233, Apr. 2021.

[3] C. Xu, C. Zhang, J. Xu, and J. Pei, "SlimChain: Scaling blockchain transactions through off-chain storage and parallel processing," *Proc. VLDB Endowment*, vol. 14, no. 11, pp. 2314–2326, Jul. 2021.

[4] K. Miyachi and T. K. Mackey, "hocbs: A privacy-preserving blockchain framework for healthcare data leveraging an on-chain and off-chain system design," *Inf. Process. Manage.*, vol. 58, no. 3, pp. 1–24, 2021.

[5] Y. Miao, Q. Huang, M. Xiao, and W. Susilo, "Blockchain assisted multicopy provable data possession with faults localization in multi-cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 3663–3676, 2022.

[6] S. Wilkinson, T. Boshevski, J. Brandoff, and V. Buterin. (2014). *Storj A Peer-To-Peer Cloud Storage Network*. [Online]. Available: http://storj.io/storj.pdf

[7] J. H. King. (2017). *Guaranteeing IoT Data Integrity With Blockchain: Rail Use Case*. [Online]. Available: https://www.ericsson.com/globalassets/digital-asset-integrity-service-user-guide_rev0726.pdf

[8] P. Golle, S. Jarecki, and I. Mironov, "Cryptographic primitives enforcing communication and storage complexity," in *Financial Cryptography*. Berlin, Germany: Springer, 2002, pp. 120–135.

[9] A. P. Mohan, M. Asfak R., and A. Gladston, "Merkle tree and blockchain-based cloud data auditing," *Int. J. Cloud Appl. Comput.*, vol. 10, no. 3, pp. 54–66, Jul. 2020.

[10] J. Mao, Y. Zhang, P. Li, T. Li, Q. Wu, and J. Liu, "A position-aware Merkle tree for dynamic cloud data integrity verification," *Soft Comput.*, vol. 21, no. 8, pp. 2151–2164, Apr. 2017.

[11] G. Ateniese et al., "Provable data possession at untrusted stores," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, New York, NY, USA, 2007, pp. 598–609.

[12] N. Dhakad and J. Kar, "EPPDP: An efficient privacy-preserving data possession with provable security in cloud storage," *IEEE Syst. J.*, vol. 16, no. 4, pp. 6658–6668, Dec. 2022.

[13] G. Ateniese et al., "Remote data checking using provable data possession," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, pp. 1–34, May 2011.

[14] J. Shen, P. Zeng, and K. R. Choo, "Multicopy and multiserver provable data possession for cloud-based IoT," *IEEE Internet Things J.*, vol. 9, no. 14, pp. 12300–12310, Jul. 2022.

[15] N. Kaaniche, E. E. Moustaine, and M. Laurent, "A novel zero-knowledge scheme for proof of data possession in cloud storage applications," in *Proc. 14th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, Chicago, IL, USA, May 2014, pp. 522–531.

[16] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. 4th Int. Conf. Secur. Privacy Commun. Netowrks*, New York, NY, USA, Sep. 2008, pp. 1–10.

[17] J. Yu and H. Wang, "Strong key-exposure resilient auditing for secure cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 8, pp. 1931–1940, Aug. 2017.

[18] A. Juels and B. S. Kaliski, "Pors: Proofs of retrievability for large files," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, New York, NY, USA, Oct. 2007, pp. 584–597.

[19] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Melbourne, VIC, Australia, 2008, pp. 90–107.

[20] Y. Yang, Y. Chen, F. Chen, and J. Chen, "An efficient identity-based provable data possession protocol with compressed cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 1359–1371, 2022.

[21] J. Xu and E.-C. Chang, "Towards efficient proofs of retrievability," in *Proc. 7th ACM Symp. Inf., Comput. Commun. Secur.*, New York, NY, USA, 2012, pp. 79–80.

[22] K. Zhu, Y. Ren, and Q. Zhu, "A provable data possession protocol in cloud storage systems with fault tolerance," in *Proc. IEEE Conf. Dependable Secure Comput. (DSC)*, Aizuwakamatsu, Japan, Jan. 2021, pp. 1–6.

[23] M. B. Paterson, D. R. Stinson, and J. Upadhyay, "Multi-prover proof of retrievability," *J. Math. Cryptol.*, vol. 12, no. 4, pp. 203–220, Dec. 2018.

[24] H. Tian, F. Nan, H. Jiang, C.-C. Chang, J. Ning, and Y. Huang, "Public auditing for shared cloud data with efficient and secure group management," *Inf. Sci.*, vol. 472, pp. 107–125, Jan. 2019.

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

[26] Y. Zhang, J. Ni, X. Tao, Y. Wang, and Y. Yu, "Provable multiple replication data possession with full dynamics for secure cloud storage," *Concurrency Computation: Pract. Exper.*, vol. 28, no. 4, pp. 1161–1173, Mar. 2016.

[27] H. Wang, Q. Wu, B. Qin, and J. Domingo-Ferrer, "Identity-based remote data possession checking in public clouds," *IET Inf. Secur.*, vol. 8, no. 2, pp. 114–121, Mar. 2014.

[28] Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren, "Enabling efficient user revocation in identity-based cloud storage auditing for shared big data," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 3, pp. 608–619, May 2020.

[29] J. Yu, K. Ren, C. Wang, and V. Varadharajan, "Enabling cloud storage auditing with key-exposure resistance," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 6, pp. 1167–1179, Jun. 2015.

[30] J. Yu, R. Hao, H. Xia, H. Zhang, X. Cheng, and F. Kong, "Intrusion-resilient identity-based signatures: Concrete scheme in the standard model and generic construction," *Inf. Sci.*, vols. 442–443, pp. 158–172, May 2018.

[31] J. Yu, K. Ren, and C. Wang, "Enabling cloud storage auditing with verifiable outsourcing of key updates," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 6, pp. 1362–1375, Jun. 2016.

[32] L. Zhou, A. Fu, G. Yang, H. Wang, and Y. Zhang, "Efficient certificateless multi-copy integrity auditing scheme supporting data dynamics," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1118–1132, Mar. 2022.

[33] Y. Ji, B. Shao, J. Chang, and G. Bian, "Privacy-preserving certificateless provable data possession scheme for big data storage on cloud, revisited," *Appl. Math. Comput.*, vol. 386, Dec. 2020, Art. no. 125478.

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.

[34] H. Wang, Q. Wang, and D. He, "Blockchain-based private provable data possession," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2379–2389, Sep. 2021.
[35] D. Francati et al., "Audita: A blockchain-based auditing framework for off-chain storage," in *Proc. 9th Int. Workshop Secur. Blockchain Cloud Comput.*, May 2021, pp. 5–10.
[36] Y. Du, H. Duan, A. Zhou, C. Wang, M. H. Au, and Q. Wang, "Enabling secure and efficient decentralized storage auditing with blockchain," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 3038–3054, Sep. 2022.
[37] Y. Li, Y. Yu, R. Chen, X. Du, and M. Guizani, "IntegrityChain: Provable data possession for decentralized storage," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1205–1217, Jun. 2020.
[38] Y. Du, H. Duan, A. Zhou, C. Wang, M. H. Au, and Q. Wang, "Towards privacy-assured and lightweight on-chain auditing of decentralized storage," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Singapore, Nov. 2020, pp. 201–211.
[39] I Corporation. (2023). *Intel Software Guard Extensions*. [Online]. Available: https://www.intel.com/content/www/us/en/developer/tools/software-guard-extensions/attestation-services.html
[40] S. T. Siddiqui, R. Ahmad, M. Shuaib, and S. Alam, "Blockchain security threats, attacks and countermeasures," *Adv. Intell. Syst. Comput.*, vol. 1097, pp. 51–62, Mar. 2020.
[41] R. Zhang, R. Xue, and L. Liu, "Searchable encryption for healthcare clouds: A survey," *IEEE Trans. Services Comput.*, vol. 11, no. 6, pp. 978–996, Nov. 2018.
[42] Y. Wang, S. Sun, J. Wang, J. K. Liu, and X. Chen, "Achieving searchable encryption scheme with search pattern hidden," *IEEE Trans. Services Comput.*, vol. 15, no. 2, pp. 1012–1025, Mar. 2022.
[43] S. Fei, Z. Yan, W. Ding, and H. Xie, "Security vulnerabilities of SGX and countermeasures: A survey," *ACM Comput. Surveys*, vol. 54, no. 6, pp. 1–36, Jul. 2022, doi: 10.1145/3456631.
[44] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. 3rd Symp. Operating Syst. Design Implement.*, New Orleans, LA, USA, 1999, pp. 1–14.
[45] S. Darbha and D. P. Agrawal, "Optimal scheduling algorithm for distributed-memory machines," *IEEE Trans. Parallel Distrib. Syst.*, vol. 9, no. 1, pp. 87–95, Jan. 1998.
[46] N. McKeown, "The iSLIP scheduling algorithm for input-queued switches," *IEEE/ACM Trans. Netw.*, vol. 7, no. 2, pp. 188–201, Apr. 1999.

Lipeng Wang received the B.S. and M.S. degrees in software engineering from Sichuan University, Chengdu, China, in 2009 and 2012, respectively. He is currently pursuing the Ph.D. degree with Peking University, Beijing, China. He has published more than 30 scientific papers, including IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and CVPR. His current research interests include information security and blockchain.

Zhi Guan received the B.S. and Ph.D. degrees in software engineering from Peking University, Beijing, China, in 2004 and 2009, respectively. He is currently an Assistant Professor with Peking University. His current research interests include cryptography.

Zhong Chen received the B.S. and Ph.D. degrees in software engineering from Peking University, Beijing, China, in 1983 and 1989, respectively. He was the Dean of the School of Software and Microelectronics, Peking University, where he is currently a Professor. He is also the Director of the Key Laboratory of Network and Software Security Assurance of the Chinese Ministry of Education. He has published more than 200 scientific papers, including ICSE, ISSTA, and ICPC. His current research interests include cryptography and blockchain. He is a fellow and the Executive Director of the China Computer Federation. He won two second-class prizes of the National Science and Technology Progress Award of China.

Mingsheng Hu received the B.S. degree in computer software from Central China Normal University in 1997, the M.S. degree in software engineering from Information Engineering University in 2005, and the Ph.D. degree in control engineering from the Huazhong University of Science and Technology in 2007. Since 1997, he has been with Zhengzhou Normal University, where he is currently a Professor. He has published more than 50 articles. His current research interests include information security and complex networks. He is an awarder of the National Science Fund of China.

Authorized licensed use limited to: Monash University. Downloaded on November 02,2023 at 14:36:47 UTC from IEEE Xplore. Restrictions apply.