Peer-to-Peer Networking and Applications (2023) 16:2850–2864
https://doi.org/10.1007/s12083-023-01570-1

FIBPRO: Peer-to-peer data management and sharing cloud storage system based on blockchain

Rui Han[1] · Yu Wang[1] · Mingfa Wan[1] · Teng Yuan[1] · Guozi Sun[1,2]

Received: 1 June 2023 / Accepted: 28 September 2023 / Published online: 7 October 2023

Abstract

With the continuous and rapid development of cloud-based data services, the total volume of Internet data is experiencing explosive growth. Nevertheless, contemporary centralized cloud storage-oriented data service providers encounter significant challenges in fully satisfying the requirements for user data security, fine-grained access control, and consistently high-performance standards. In this paper, we propose and implement an efficient peer-to-peer data storage and sharing system to address the challenges faced by service providers. Our solution utilizes the modified EOSIO blockchain and the InterPlanetary File System (IPFS) distributed storage as the underlying data storage framework. We employ hybrid encryption to ensure the secrecy of the shared data of the users and also to facilitate multiple uses and persistent storage of the shared data of the users. Moreover, with the expansion of the blockchain component, we provide a flexible transaction information audit solution that helps to trace the source of malicious behavior and reduces the cost of using blockchain information. In system analysis and experimental evaluation, compared to traditional blockchain storage, FIBPRO has theoretically achieved a 98.76% reduction in on-chain storage consumption. In practical concurrency testing, it achieved a comprehensive performance of approximately 1300 TPS (transactions per second), with an average upload efficiency of about 2.31MB/s and a download rate of about 5.29MB/s. These results demonstrate the system's availability and scalability.

Keywords Distributed storage · Data sharing · Blockchain · Information security · Custom audit

1 Introduction

Under the accelerating evolution of information technology, cloud storage is gradually replacing traditional monolithic local database solutions as it offers benefits such as ease of scaling, flexible access, and low-cost data disaster recovery backup [1, 2]. Cloud-based data management has become increasingly popular, yet it faces new challenges in the age of metadata. Due to the centralization of geographic location and the centralized physical media of cloud storage, failures of the physical media on which data is stored can lead to data corruption and loss [3]. Meanwhile, for some sensitive data, such as medical data, payment bills, and confidential documents, the cloud provider may use them for commercial purposes without the consent of the holder. When there are disputes among stakeholders such as data owners, data holders, and data users, centralized cloud storage facilities can lead to unfair disposal and corruption.

With the implementation of blockchain technology, fully addressing the challenges of cloud storage has become feasible. Blockchain technology was first implemented by Satoshi

This article is part of the Topical Collection: *3 - Track on Blockchain*
Guest Editors: Haojin Zhu

✉ Guozi Sun
sun@njupt.edu.cn

Rui Han
1021041506@njupt.edu.cn

Yu Wang
1221045515@njupt.edu.cn

Mingfa Wan
1221045514@njupt.edu.cn

Teng Yuan
1321048416@njupt.edu.cn

[1] School of Computer Science and Technology, Nanjing University of Posts and Telecommunications, WenYuan Road, Nanjing 210023, JiangSu, China

[2] Institute of Data Security & Compliance, Nanjing University of Posts and Telecommunications, Nanjing, China

Nakamoto in his famous white paper in a real-world application of the Bitcoin cryptocurrency [4]. Blockchain forms a decentralized data ledger using a special data structure and a distributed network node configuration, which helps data to be hosted more securely [5, 6]. This promises breakthrough solutions in privacy data management, multi-party data sharing, and other permission policy-related activities. The key features of blockchain, such as decentralization, invariance, auditability, data traceability, privacy, and security, may help the data storage industry explore more advanced solutions [7, 8]. The primary obstacle to the widespread adoption of blockchain technology is its poor performance, which constrains its integration with specific applications.

Despite all these conveniences, there are additional issues with data sharing and storage. Unlike specific types of documents such as archives and medical data, private p2p data management typically needs to cope with higher frequency access requests and diverse data types [9]. Moreover, unlike public data, private data processing requires ensuring confidentiality throughout the data content. In general, data sharing and storage face the following challenges.

1. Efficient System Performance: Data management systems need to handle high performance, high availability, and scalability requirements to cope with different user requests.
2. Data Security & Unauthorized Access: Unauthorized data attackers may gain access to the system to snatch data. In this case, the aggressor forges and uses the user's identity information, which will lead to data leakage and tampering, reducing the privacy of the data.
3. Information Tracking: Data hosting and delivery records need to be secured and maintained to meet complex data-sharing requirements. For example, the exploration of data-sharing records in the event of a data breach.

To address the above requirements, in this paper, we construct a blockchain-based data management and sharing with a primary focus on peer-to-peer data, and propose an open-access custom audit model to address security issues. The main contributions of this work are fourfold:

- Streamlined Data Sharing Process: We optimized the transaction flow in data sharing by reducing the number of actual content passes to help improve the overall quality of data sharing.
- Blockchain Network with High Performance: We built an efficient blockchain operating environment with FIBOS[1] and the transaction execution process is optimized by on-chain and off-chain collaborative transactions to implement a data processing system for a medium-sized web application (thousands of concurrency).
- Open Access Custom Audit: We design customized audit strategies for different roles in data transactions to obtain information of more interest to diverse users.
- Data Analysis and Experiments: We tested the performance of the system in terms of storage space and time consumption for heterogeneous formats of data with different sizes.

We have organized the existing papers below. On the basis of a literature review, Section 2 introduces related work on data storage. Section 3 gives the framework fundamentals of blockchain. Section 4 describes the general working model of the system operation. Section 5 presents the data access approaches in our system. Section 6 describes the details of the transaction model. Section 7 describes the storage security and privacy protection of the system and presents the design principles of the audit module. Section 8 gives an assessment of the energy consumption of the message calculation and transmission of the FIBPRO scheme.

2 Related work

Some of the initial research carried out by scholars studying blockchain applications have focused on the combined use of blockchain and cloud storage [10]. Chen designed a storage scheme to manage personal medical data based on blockchain and cloud storage [11]. Mustafa's methods focus on the qualities that influence customers' interest in and approval of blockchain technology in cloud storage [12]. So tighter and more advanced security requirements have been provided to increase the protection of cloud-outsourced healthcare data.

While the overall framework using a combination of blockchain and cloud storage proves to be feasible and scalable, the use of cloud service agents makes the system too centralized. Considering the trust factor of the system's third party (e.g., cloud service provider) the security of the system is challenged when the service provider cannot be fully trusted. In response to this problem, Nizamuddin proposed an IPFS-based solution and framework for document sharing and version control to facilitate multi-user collaboration and track changes in a trusted, secure, and decentralized manner, with no involvement of a centralized trusted entity or third party [13]. The change of storage medium solves the potential problems of data tampering and data loss in the use of the centralized database. However, the method of direct archiving by IPFS lacks access control for user identity, which will lead to unauthorized access of data [14, 15].

[1] FIBOS is a customized modification of EOSIO that extends on-chain governance and cross-chain interactions. https://dev.fo/en-us

According to the security requirements of different scenarios, researchers put forward a variety of improvement methods. A series of research has been carried out on the verification strategy of system user identity information. Some propose dynamic identity management and verification strategy [16], through the variable user rights management, which can improve the flexibility of data access and the security of data content. Further, usage of zero-knowledge proof technology enhanced the security of identity verification [17]. By reducing the amount of user information provided during validation, the exposure of information and possible attacks are reduced.

Other scholars focus on the challenges of combining signature technology with blockchain to improve the recording and sharing of private data on the blockchain. One of the schemes using ring signature and group signature achieves the privacy information protection of blockchain by hiding the information of the signature user [18, 19]. The use of a blind signature scheme is independent of the signature link and the message content, also to achieve the protection of the blockchain information.

The blockchain technology and cryptography combination has greatly improved the security of the system, but due to the performance bottleneck of the blockchain, the complex cryptography will lead to a further decline in the usability of the blockchain [20]. Consensus algorithm improvement, storage structure optimization, and off-chain storage expansion optimize system execution from two dimensions: transaction execution time optimization and storage capacity [21].

3 Blockchain framework fundamentals

Blockchain technology has undergone frequent updates and iterations in recent years. The replacement of consensus algorithms and smart contract technology has led to significant adjustments in the architecture of blockchain technology. A depiction of the current mainstream blockchain architecture is presented in Fig. 1.

3.1 Consensus algorithm

In blockchain technology, the consensus algorithm plays a crucial role in ensuring the orderly and equitable functioning of the decentralized system, as well as determining the processing logic of network transactions [22]. The implementation methodologies of diverse consensus algorithms vary significantly, leading to discernible disparities in their operational efficiency. The mainstream consensus algorithms can be broadly classified into two major categories [23]:

Verification-based consensus algorithms, such as PoW, which reach consensus by waiting for results to be calculated and verified. This type of algorithm has the problem of forking, which means that if multiple nodes verify successfully at the same time, it will take time for the network to resolve the fork, resulting in problems such as waiting for confirmation and too long time taking of transactions [24].

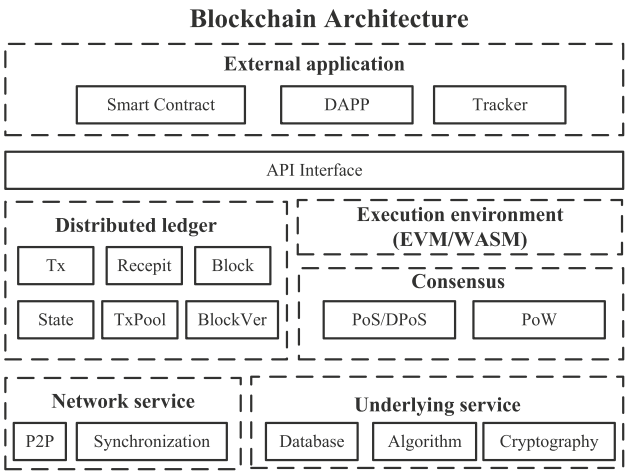


Fig. 1 Blockchain mainstream architecture

Voting-based consensus algorithms, such as the DPoS (Delegated Proof of Stake) used in this model, in which participants elect a few representative nodes to operate the network, using professional web servers to ensure the security and performance of the blockchain network. Consensus reaching in the DPOS mechanism does not require solving mathematical puzzles by consuming arithmetic power; instead, participants elect block-producing nodes, which then collaborate and take turns in block manufacturing [25, 26]. If the producer is incompetent, there is always the possibility of being voted out, which alleviates the performance problem of the POS. Table 1 gives a comparison of several consensus algorithms.

3.2 Smart contracts

Smart contracts refer to computer programs that are designed to execute and enforce the terms of a contract in an automated and autonomous manner. These contracts are defined by lines of code and enforced through a decentralized network of computers, without the need for intermediaries or human intervention [27]. In FIBPRO, we used WebAssembly(WASM) to build the execution environment of smart contracts and intended to reduce the dependence on centralized servers [28].

3.3 Tracker

Tracker typically refers to the component of a blockchain network responsible for tracking the status of nodes to

Table 1 Comparison of consensus algorithm

Consensus algorithm	DPoS	PoS	PoW
Representative Chain	EoS	ETH	BTC
Number of Nodes	Fewer	Fewer	More
Outgoing Block Nodes	Super nodes	Validators	Mining pool nodes
Block Output Efficiency	0.5 seconds	12 - 15 seconds	10 minutes
Block Interval	Fixed	Fixed	Adjusted every 14 days
Block Probability	Generated in a fixed order	By equity weighting	By arithmetic power
Irreversible Confirmation	360 blocks of 180 seconds	Average 3.9 minutes	Around 6 blocks 1 hour
Governance in the Chain	Vote	Vote	None

enable communication and information collection between network nodes.

We designed and implemented a more aggressive tracker to analyze and disassemble the content of the packaged blocks. The hash check structure of the Merkle tree is used to realize on-chain content check, and the comprehensive cost of system execution in a smart contract is reduced by off-chain extension of on-chain contract [29]. Our proposed custom audit approach also relies on a tracker to collect and filter information in the blockchain.

4 System model

The working process of this system is done with a collaborative utilization of IPFS and blockchain, and a brief working model of the system is visually depicted in Fig. 2.

4.1 User model

The user model comprises two primary components: authentication and data transactions. Upon a user's initial access to the system, the network generates unique public key p_u and private key s_u, which are respectively stored by the blockchain and the user. To ensure robust security measures, the system utilizes two-factor verification to enhance the security level. Specifically, during user login, both the login password pw_u and the asymmetric secret key p_u are verified to authenticate the user's identity.

Furthermore, to safeguard the confidentiality of the user's data, the system encrypts the data M_{ori} using the public key p_u before uploading it to IPFS. Then, IPFS networks use sharding technology to copy and store encrypted data M_{mod}. During the receiving transaction processing, the user is required to decrypt the relevant information using their private key s_u when accessing the data, which decrypts the original shared data M_{ori}.

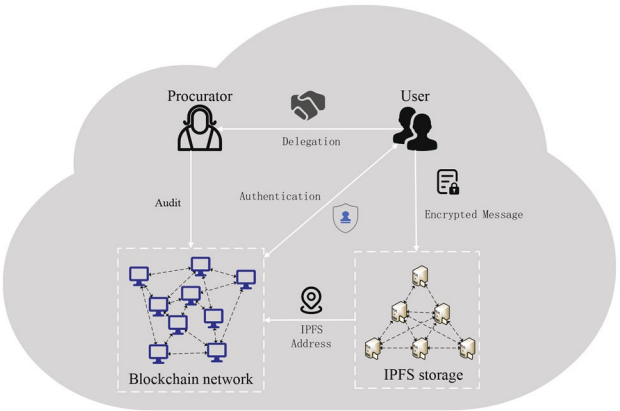


Fig. 2 System working model

4.2 Blockchain node model

The blockchain network uses a DPoS-based consensus method to achieve distributed transactional consistency. We set up a total of 12 delegate nodes in the blockchain network to alternate the block issuance activities, which can reduce the blockchain system operation time compared to the preset number of nodes of EOSIO or FIBOS. Since blockchain delegate nodes need to be responsible for transaction generation, transaction broadcasting, and block validation, individual data owners and users are not fully competent. In the blockchain node design of this system, IPFS service providers and responsible data user groups are considered and are pre-defined as candidate delegate nodes.

4.3 IPFS node model

The IPFS data storage network is the primary storage medium for the entire distributed data management system, holding unstructured user private data. This data is encrypted before it is submitted to the blockchain network, and the information is hashed and submitted to the blockchain as an index for data queries to be saved and shared [30]. Even if illegal information visitors appear to have accessed the encrypted data directly from the IPFS data store via the IPFS address, they will not be

able to decrypt the data to obtain any useful information. With blockchain-generated and stored user secret keys, IPFS information storage has additional security [31]. Meanwhile, the distributed and decentralized design of IPFS itself allows the system to store data without being constrained by operational efficiency and scalability issues.

4.4 Open access blockchain audit and verification model

The blockchain service is built in an open distributed network environment, so the information about the data stored in the blockchain is available to all blockchain nodes. Any responsible or curious blockchain member can become a full node of the blockchain, and monitor and audit all transactions on the blockchain through *Tracker*. Of course, no private data of the user is kept in the blockchain. What is mainly recorded in the blockchain is some structured data and a record of the system operation and activities. The inspector needs to cooperate with off-chain information to realize the audit of system user behavior and data information.

5 Data access approaches

Data access approaches implemented using cloud storage or delegated to traditional centralized data service providers do not fully guarantee the data owner's access control over the data. An intermediary in the data access model may bypass the data owner and grant access to the data to a third party. Fortunately, data access control through blockchain smart contracts for storage information retrieval, user authentication, and access authorization can effectively alleviate this problem. For the data users (sharers and receivers), they don't care what encryption algorithm the system uses or how the system is linked to the blockchain. They can use a traditional web front-end page to control and authorize access to the data, just as they would with a traditional centralized data system.

5.1 Overview

Any data uploaded by users for storage and sharing will be encrypted and stored in the IPFS data network after entering the system. The data owner first generates the symmetric secret key and submits the encrypted data to the IPFS network for storage, and stores the data hash index and address information back from IPFS in the blockchain network, as shown in the information upload data flow diagram. When users share data, they need to submit a data exchange transaction in the blockchain network. The data exchange transaction does not directly transmit the data itself. Since the data entity is stored in the IPFS network, Once the symmetric key sharing is completed, the receiver can access the data content. The sharer encrypts the symmetric secret key of the data to be shared with the recipient's public key and packages this shared signature to the blockchain network. The receiver who obtains permission decrypts the secret key information and accesses the storage network through the IPFS address, downloads, decrypts, and assembles the data finally submits the record of the received transaction in the blockchain so that the blockchain network can audit the relevant behavior, the whole process is shown in the Fig. 3.

5.2 Blockchain transaction approaches

In this data management system, most of the structured data is defined and stored in the blockchain, which not only stores this data but also records the key transaction information. There are four main types of transactions in the system: user authentication transactions, data indexing transactions, data sharing transactions, and data download transactions.

- **User authentication transactions** - This type of transaction verifies user login information by cryptographic methods and also logs abnormal login behavior to ensure user security.
- **Data upload transactions** - By submitting an index of the encrypted data kept in the IPFS network, the system can share data in the future by querying the information on the blockchain.
- **Data sharing transactions** - After the data sharer has locally processed the secret key required for data sharing, the relevant information will be signed and packaged into a transaction.
- **Data download transactions** - After decrypting the shared secret symmetric secret key information and obtaining the IPFS file address, the data recipient finishes decrypting and assembling the file locally, then signs this reception and submits the blockchain transaction.

6 Proposed transaction mechanism

The data protection and sharing function realized by this system ensures the safety of data through the security features of blockchain and the cryptography method. Private data submitted to the system will be turned over to the IPFS network after AES/SM4 symmetric encryption. Then, these data will be transmitted within transactions via the blockchain to the receiving entities. In this transmission module, the asymmetric key of the receiver is used to ensure the security of the shared data. Moreover, Users need to sign the transactions to ensure the transactions are valid and originating from authorized users. The algorithms used for encryption, signature, and hash check are shown in Table 2.

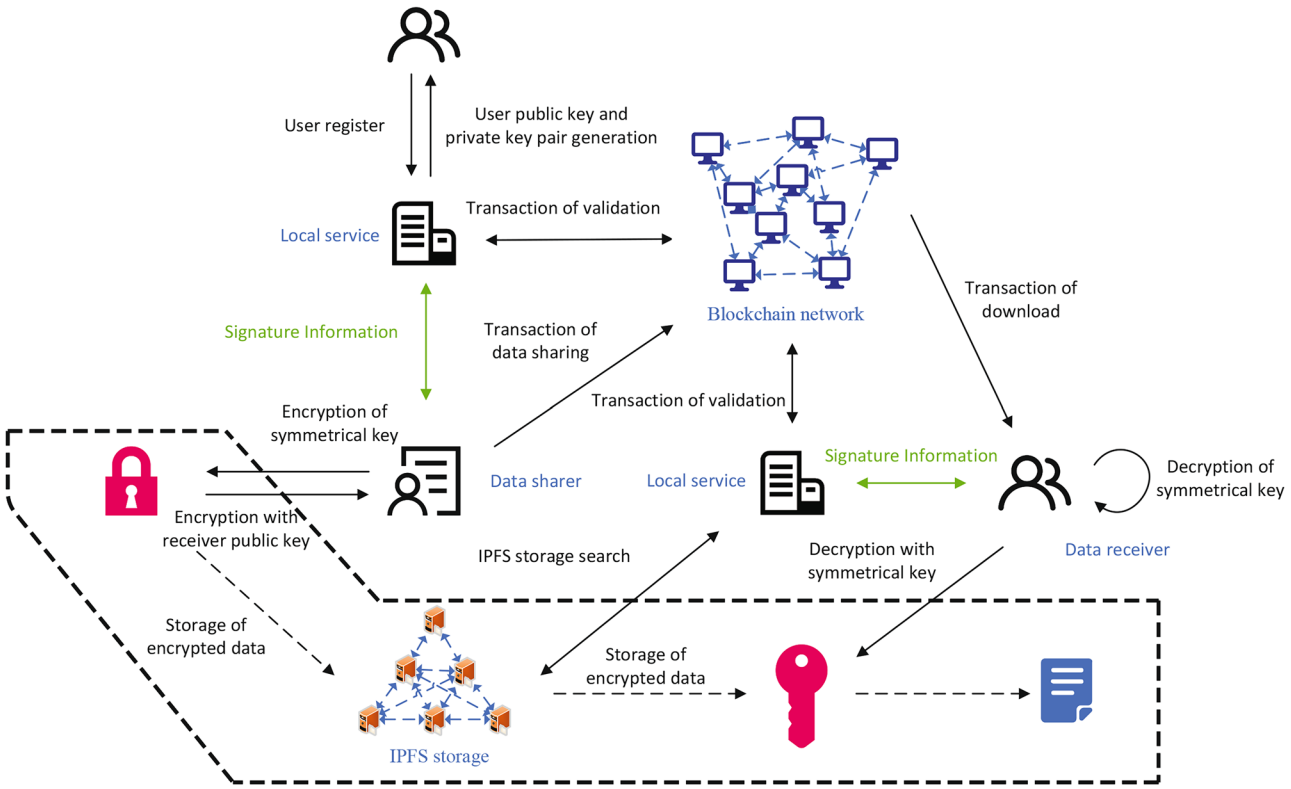


Fig. 3 Data storage and sharing model

6.1 Data upload transactions

The user data managed by the system is generated by users who host the original data. Before the private data flows into the system, it is encrypted on the user side through the independent symmetric secret key of each file generated by the user. The encrypted data will then be handed over to the IPFS network for hosting, while the index information generated by IPFS will be packaged and submitted into a blockchain transaction along with the encrypted data secret key for data access and sharing. The upload flow chart is shown in Fig. 4. The information involved in the data upload transaction is as follows:

- Uploader's public key p_u
- Symmetric encryption key k_s for submission data

Table 2 Cryptographic algorithm usage

Cryptographic techniques	Algorithm
Symmetrical encryption	AES/SM4
Asymmetrical encryption	ECC/SM2
Signature	ECC/SM2
Hash	SHA-256

- Encrypted message key $k_s^{'}$ which encrypted k_s with the uploader's public key
- Global time as Timestamp t_{up}
- Hash of encrypted data present in IPFS storage will act as the IPFS storage address $ipfs_h$
- Signature $Sign_{up}$ is generated from the IPFS storage address and timestamp through the uploader's private key

Among them, p_u is already known by the system and k_s should not be packed into the Transaction Because it is confidential. $k_s^{'}$, t_{up}, $ipfs_h$ and $Sign_{up}$ functioning as transaction parameters are handled to the smart contract SC_{up} . The transaction is generated as shown in the formula.

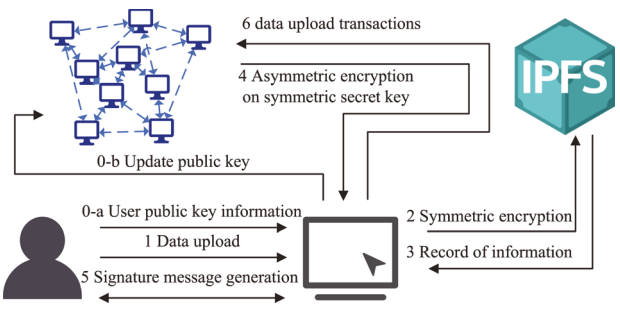


Fig. 4 Upload data flow

$$T_{up} = SC_{up}\left(k_s{}', t_{up}, ipfs_h, Sign_{up}\right) \tag{1}$$

$$ipfs_h = H(Enc_{k_s}(data_{up})) \tag{2}$$

$$k_s{}' = Enc(p_u, k_s) \tag{3}$$

$$Sign_{up} = Sign(H(t_{up}, ipfs_h)) \tag{4}$$

6.2 Data sharing transactions

Data sharing is achieved through symmetric key messaging for data content access. The initial step involves a proactive attempt to access the symmetric key information from the local cache, thereby optimizing operational efficiency and minimizing reliance on frequent interactions with the blockchain network. If this attempt yields the desired key information, the subsequent stages of the process are streamlined, bypassing the blockchain query. However, in cases where the local cache retrieval proves unsuccessful or the cached information has expired, the system proceeds to access the required key data from the blockchain, a repository where this information was initially recorded during the data upload process. Subsequently, the sharer utilizes their private key to decrypt the data secret key, simultaneously acquiring essential public key details of the intended recipient. Then the data secret key is subjected to asymmetric encryption using the recipient's public key, generating a secure key-sharing message, thereby ensuring that only the designated recipient can decrypt and access the shared data. Finally, the sharer signs the data information and the cryptographic key and packages the relevant information into a transaction for submission to the blockchain network. The data sharing flow chart is shown in Fig. 5. The information involved in the data-sharing transaction is as follows:

- Uploader's public key p_u
- Uploader's private key S_u
- Recipient's public key p_r
- Encrypted message key k_s' which encrypted k_s with the uploader's public key
- Encrypted message key k_r which encrypted k_s with the receiver's public key
- Global time as Timestamp t_{sh}
- Signature $Sign_{sh}$ is generated from the key of the data, timestamp through uploader's private key

Among them, p_u and p_r are already known by the system and can be get from the blockchain directly. k_s' is also available through the retrieval of sharing information uploaded before. Data sharing is achieved through the decryption of k_s' and the encryption to generate k_r to safely transfer the secret key. k_r, t_{sh} and $Sign_{sh}$ functioning as transaction parameters are handled to the smart contract SC_{sh}. The transaction is generated as shown in the formula.

$$T_{up} = SC_{up}\left(k_r, t_{sh}, Sign_{sh}\right) \tag{5}$$

$$k_s = Dec(s_u, k_s{}') \tag{6}$$

$$k_r = Enc(p_r, k_s) \tag{7}$$

$$Sign_{up} = Sign(H(t_{sh}, k_r)) \tag{8}$$

6.3 Data download transactions

Data download is the final step in data transmission. The receiver gets the encrypted message delivered by the data sharer through the transaction information in the blockchain. So, the symmetric secret key information can be restored by the recipient's private key. After that, the encrypted data content can be accessed in the IPFS storage by querying the IPFS storage address of the data recorded in the blockchain. The combination of the two gives the initial state of the data. Finally, the recipient signs and submits the corresponding information to the blockchain network, completing the entire download process. The data download flow chart is shown in Fig. 6. The information involved in the data download transaction is as follows:

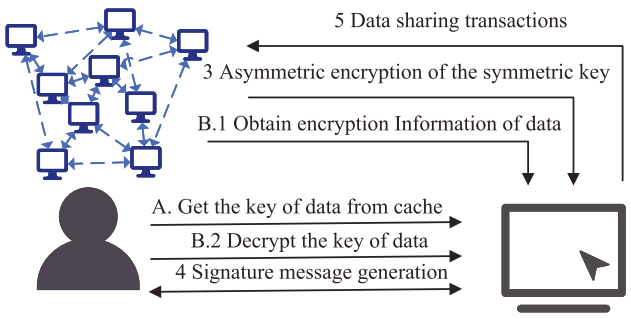


Fig. 5 Share data flow

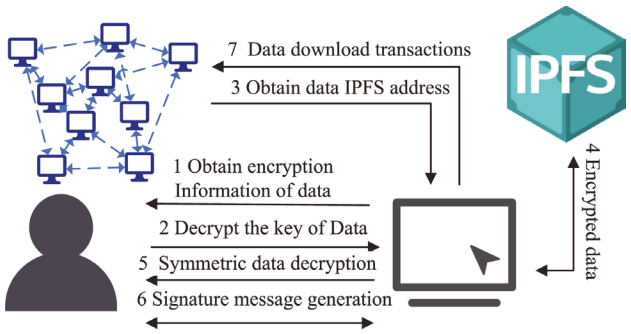


Fig. 6 Download data flow

- Recipient's private key s_r
- Encrypted message key k_r which encrypted k_s with the receiver's public key
- Global time as Timestamp t_{down}
- Hash of encrypted data present in IPFS storage will act as the IPFS storage address $ipfs_h$
- Signature $Sign_{down}$ is generated from the key of the data, timestamp through uploader's private key

Among them, k_r and $ipfs_h$ are already known by the system and can be got from the blockchain directly. And s_r is stored by the recipient locally. Data download is achieved through the decryption of $k_s^{'}$ and acquisition of data from $ipfs_h$. k_r, t_{do} and $Sign_{do}$ functioning as transaction parameters are handled to the smart contract SC_{do}. The transaction is generated as shown in the formula.

$$T_{down} = SC_{down}\left(k_r,\ t_{down}, Sign_{down}\right) \tag{9}$$

$$k_s = Dec(s_r, k_r) \tag{10}$$

$$data_{down} = Dec_{\ ks}(Enc_{k_s}(data_{up})) \tag{11}$$

$$Sign_{down} = Sign(H(\ t_{down}, k_s)) \tag{12}$$

7 Security analysis and customized audits

7.1 Security analysis

7.1.1 Storage security

IPFS, as the basic distributed storage framework in FIBPRO, achieves secure and persistent data storage in the context of file-sharding through a file chunk exchange algorithm based on the BitTorrent protocol, coupled with the single-point file information pinning services [32].

Within FIBPRO, we establish a minimum file pinning threshold as a fundamental measure for ensuring data security. This serves as an effective safeguard, particularly for less frequently accessed data. In the case of high-frequency access data, IPFS's file-sharding algorithm significantly enhances node caching, thereby ensuring the security of data storage.

7.1.2 Privacy security

The pure IPFS storage network does not inherently provide means for data privacy protection and data encryption. In FIBPRO, we have implemented symmetric encryption at the ingress point for both storage and sharing of data, followed by the operation of storing encrypted data shards. In the subsequent process of sharing data among different entities, we utilize a data key delivery method based on the asymmetric key of the receiving entity to ensure privacy during the data-sharing process.

By relying on cryptographic algorithms, we address the absence of inherent data protection mechanisms in the IPFS network. Additionally, leveraging the transactional mechanisms of blockchain, FIBPRO offers data operation records for stateless IPFS storage, facilitating operation tracing and information auditing. This significantly increases the difficulty and risk of malicious data operations, thereby enhancing data privacy security to a certain extent.

7.2 Customized audits

Blockchain is designed and implemented using immutable technology and distributed ledger technology, which makes the data in the blockchain uniformly monitored and tamper-proof by the entire blockchain network. This perfectly meets the requirements for data authenticity and integrity in auditing, while the process of accessing blockchain information through the user's private key also meets the security requirements, reducing the possibility of misstatement and deliberate provision of false accounting data by the audited entity. Up to now, many organizations and individuals have carried out information auditing work for public blockchain networks such as Bitcoin and Ethereum [33–35], and they mainly identify abnormal transactions such as circular transfers and falsified transaction quantities through analysis of transaction behavior [36].

However, this information-synchronized audit method for all blockchain transactions is quite expensive. Take our data synchronization of the goerli test chain as an example. It took us 5 days to synchronize the data of the main chain of about 600G on the cloud server. Therefore, in our FIBPRO system, we designed a customizable audit module that allows information auditors to quickly access their curious information content and audit data, and rebuild a local copy of the data through the tracker component to facilitate their subsequent further access and analysis.

Figure 7 shows the working process of the audit module. First, through blockchain authentication, the auditor enters the blockchain network to obtain block information. Second, by splitting the block content, the auditor obtains the various transaction information packaged and integrated by each block. Then, through locally customized transaction filters, the auditor can quickly access the traders and transaction contents that are of interest to the auditor. In the case of FIBPRO, filter criteria are in most cases set to specific users or shared files to enable auditing of critical information.

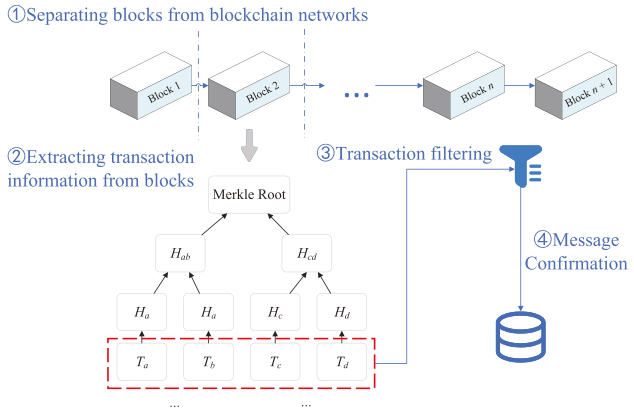


Fig. 7 Audit model

Since the main work of the audit module is accomplished off-chain, the auditor has more flexibility in data processing. By splitting the block content, the auditor can access the entry parameters of the contract and use the same execution logic as the on-chain contract to obtain the equivalent results as the blockchain state database. In addition, auditors can also process incoming parameters through custom logic to build their offline data structures which is similar to the event component in Solidity, the programming language of the Ethernet network. By synchronizing recorded operational activities on the blockchain, auditors can easily correlate them with actual requests, thus discerning the correctness of actions and identifying potentially threatening anomalies, thereby enhancing system security.

8 System experiments and performance analysis

The performance analysis of the whole system is performed by testing the upload and download performance of various files of different sizes through experiments, mainly focusing on the evaluation of the space footprint and time consumption of the system. We performed our experiments in an experimental environment consisting of five computers. Four of the machines are used as service clusters for IPFS and FIBOS, and the last machine takes on the responsibility of file sharer and receiver respectively just as shown in Fig. 8. The server machine is running on Centos 7.9 and the client machine is running on Windows 10. The server machine is configured with an Intel Xeon E-2225G processor, the guest file service IPFS uses version 0.4.23, and all components of the client are written using Node10.19.0.

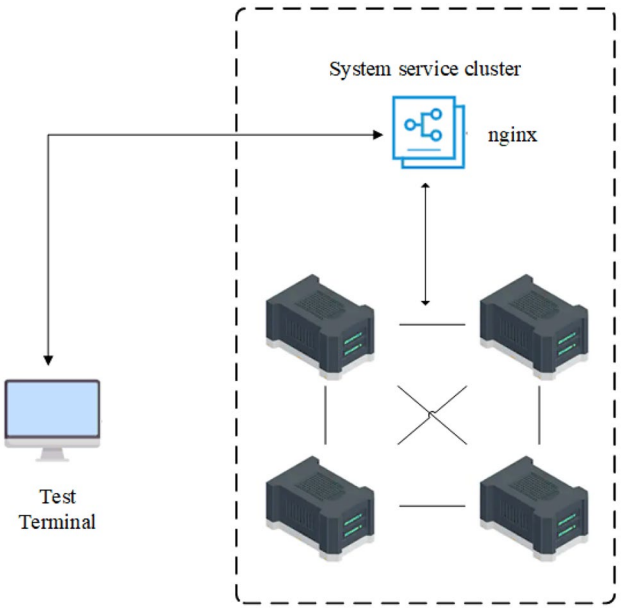


Fig. 8 Testing model

8.1 System storage analysis

8.1.1 Block capacity

The volume of blockchain transactions for file operations is analyzed statistically in order to compare the blockchain capacity consumption of on-chain and off-chain models. Using the number of blocks generated per 100 blockchain packets as an interval criterion, we analyzed the actual transaction volume contained in the number of blocks from 100 to 1000. We assume an average data file size of 25 KB. A single block contains a block header size of 257 bytes, and using the IPFS-based off-chain storage model, the size of on-chain transaction data occupied by a single data file message includes 269 bytes of secret key information and 47 bytes of IPFS address information.

The maximum capacity of a single block of the blockchain set in the blockchain network configuration is 1 MB, so 100 blocks can theoretically hold about 4096 transactions in the on-chain model, and 100 blocks can theoretically hold 324,051 transactions in the off-chain model. We calculate the number of transactions that can be accommodated by different block counts in a similar manner and compare them in the figure. Figure 9 illustrates that a single block in the off-chain storage model can hold a larger number of transactions compared to a single block in the on-chain storage model.

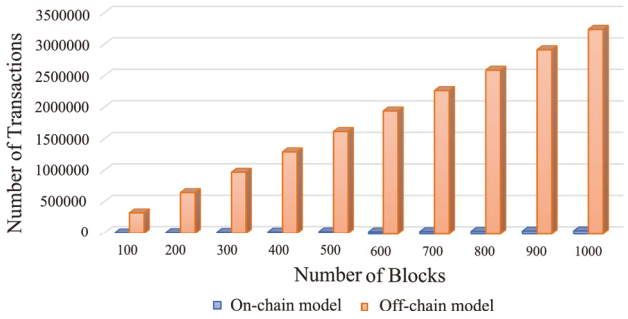


Fig. 9 Block capacity

8.1.2 Data capacity

In the analysis of data capacity, the comparison graph in the figure below was obtained by comparing the amount of data consumed in the blockchain between the on-chain and off-chain models, based on 1000 transactions and increasing it to 10,000 in regular increments of 1000 at a time. Considering the average data file size of 25KB, 1000 transactions may require 25600000 bytes in the on-chain model, while in the off-chain model, only 31600 bytes are required. A specific comparison is shown in Fig. 10.

8.1.3 Theoretical storage compression ratio

The designed off-chain storage model has considerable advantages in terms of blockchain network storage space consumption problems and storage scalability. Consider the entire data upload transaction in the data storage and sharing system, where the data information and retrieval information is stored in the blockchain, while the data itself is stored in IPFS.

In this case, the content stored by the blockchain system is the sum of the combined size of all committed transactions in the network and the size of all block headers. The size of the block headers will be the standard 257 bytes, and we assume an average data file size of 25 KB. The size

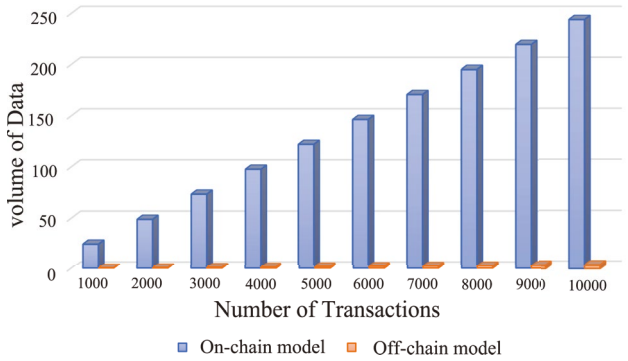


Fig. 10 Data capacity

of a data transaction using off-chain storage will be only 316 bytes. We analyze the storage compression ratio and rate evaluation by comparing the on-chain and off-chain models for each of the 1000 transactions by analyzing the data as follows.

Transaction size using on-chain storage model $tsize_{on} = 25600Bytes$

Transaction size using on-chain storage model $tsize_{off} = 316Bytes$

Log size(for audit) using on-chain storage model $tsize_{log} = 289Bytes$

Number of blocks in the blockchain in the off-chain storage model n_{off}

Number of blocks in the blockchain of the on-chain storage model n_{off}

Block head size $tsize_{log} = 257Bytes$

Storage compression ratio src:

$$src = \frac{\left(n_{off} * h_{size}\right) + \sum_{1}^{n_{off}} \left(tsize_{off} + tsize_{log}\right)}{\left(n_{on} * h_{size}\right) + \sum_{1}^{n_{on}} tsize_{on}} \tag{13}$$

The storage compression ratio is calculated with 1000 data storage transactions at a time. In our model, on-chain storage for 1000 transactions consumes 25 blocks, while off-chain storage can be focused on a single block. Therefore, we calculate a Storage compression ratio of 0.0236.

The compression ratio can be calculated by finding the savings in blockchain storage space for the off-chain model compared to the uncompressed size of the on-chain model and we calculated a compression rate of 98.76% with the Eq. (14).

$$cr = \frac{\sum_{1}^{n_{on}} tsize_{on} - \sum_{1}^{n_{off}} tsize_{off}}{\sum_{1}^{n_{on}} tsize_{on}} * 100\% \tag{14}$$

The above calculations of storage compression ratios and compression ratios for 1000 data show significant cost savings in terms of blockchain network space. Thus, a blockchain infrastructure for a data file application access model can be implemented without any scalability or storage limitations.

8.2 Analysis of transaction time consumption

8.2.1 Blockchain performance analysis

This system implementation is based on the FIBOS blockchain, which is an open-source blockchain platform based on DPoS consensus. Compared with the EoS blockchain

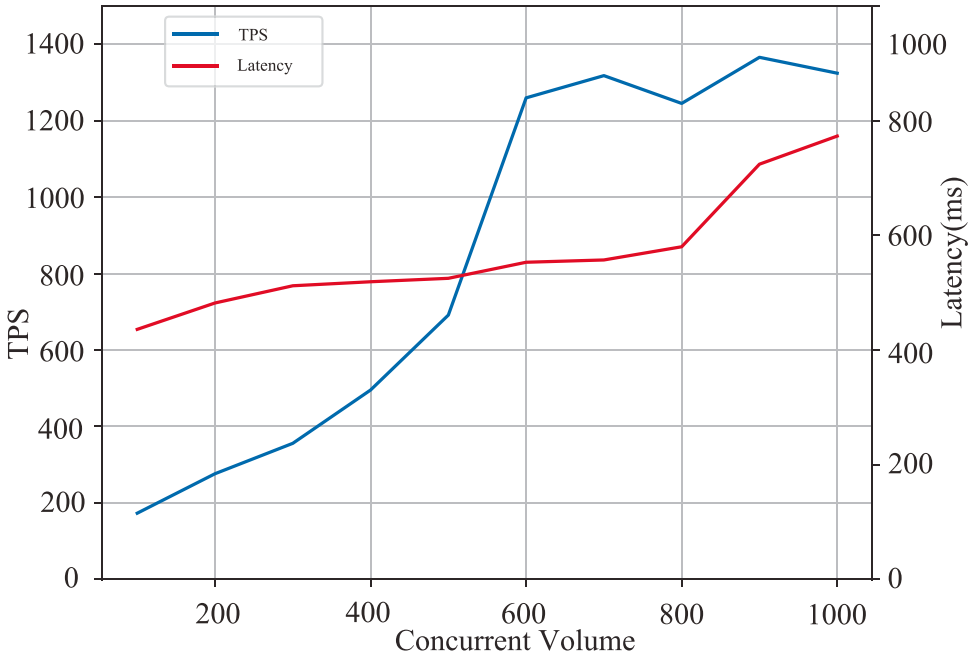


Fig. 11 System concurrency testing

platform proposed earlier, FIBOS provides language support for Node.js, which reduces the difficulty of contract development and reduces the overall project development cycle. At the same time, FIBOS provides off-chain tracker tools, which can create a collaborative off-chain database of on-chain data and reduce on-chain transaction execution overhead.

The experiments were conducted using the wrk concurrency testing tool on four server service clusters built and managed by nginx. In the wrk parameter settings, the number of simulated threads t=5, the duration of the test d=5s, and the number of simulated connections c from 100 to 1000 in increments of 100 each time 10 groups were tested. The results are shown in Fig. 11.

Obviously, when the number of concurrency increases, the system response latency grows, while the TPS gradually stabilizes when it grows to 1300. Although the TPS metrics from the system tests dwarf the thousands of TPS of the EOS platform, which is also based on DPoS consensus. However, it is worth noting that the TPS data tested here include the logical functions of the system, which is different from the TPS of simply evaluating blockchain performance.

8.2.2 Transaction time consumption analysis

The function of data transmission in this system includes the comprehensive call of many modules, such as data encryption and decryption, IPFS data upload and download, and insertion query on the chain. To observe the proportion of time consumed in each part of data transmission, we selected three main data functions of the system for performance testing. By testing multiple groups of data with different sizes (1kb, 100kb, 10mb, and 100mb), we selected the average execution time for time-consuming analysis through statistical methods. Figures 12, 13, and 14 show the time-consuming relationship between data uploading, sharing, and downloading.

The process of data uploading includes the time encrypting data by AES scheme, uploading data to IPFS, adding file information on the chain, and adding logs on the chain. As

Fig. 12 Upload transactions

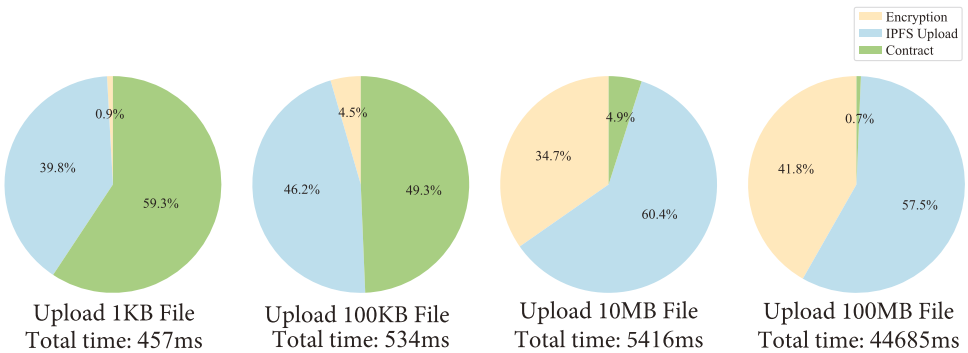
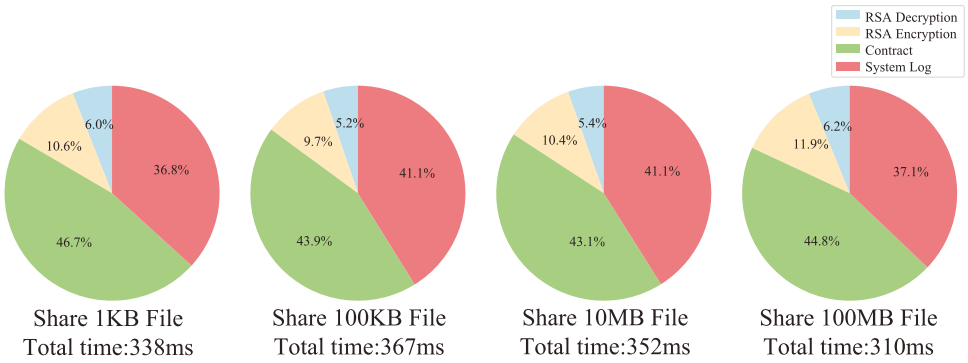


Fig. 13 Sharing transactions



expected, during the data upload process, with the increasing size of the data, the time taken to encrypt the data and upload the data to the IPFS gradually increases. Additionally, the time allowed by the smart contract, such as adding information on data and adding system logs, has nothing to do with the size of data, so the time consumption has almost no change, resulting in a relative decrease in the proportion in the figure.

The data-sharing process includes the time of decryption of the Symmetric key, the time of AES encryption of the Symmetric key with the receiver's public key, the time of adding sending records and logs to the chain, and the time of the system framework response. In the process of data upload, the data encryption time is much longer than the time of adding logs on the chain, so we omit the time of adding logs on the chain in Fig. 13. As shown in the figure, the size of the data has little impact on the time consumption of each stage of the data sharing, because the file sharing does not involve the data uploading previously, but adds a transmission record. The encryption and decryption part involved is also the encryption and decryption of the Symmetric key of data, not the file itself, which is an embodiment of our system security.

The data downloading process includes the time of downloading the encrypted data from IPFS, decrypting the Symmetric key, decrypting the data using the Symmetric key, and inserting the log on the chain. The process on the chain in this function only involves inserting logs. When uploading 1kb data, because of the small size, it takes relatively little time to download from IPFS and decrypt the file content. The main proportion of time is log insertion. As the file size increases, the proportion of time spent decrypting the file increases, because of the decentralized nature of IPFS.

When the file is uploaded to IPFS, it will be randomly cut into multiple small fragments, and each fragment will be encrypted with the sha256 algorithm, and then each fragment will be transmitted to a server. So the files uploaded by IPFS are stored in each distributed node. When downloading files, they are not downloaded from a single node, but from multiple nodes at the same time. At the same time, instant restore is realized at the terminal, which ensures that the entire file is complete only at the sending and receiving ends, and other processes are in the form of encrypted fragments, thus ensuring the security of our data. In addition, the transmission speed of the entire file has also been greatly improved, because the workload remains the same as that of the original server. Now many servers work together, greatly improving the efficiency. The built-in hash fault tolerance and hash deduplication technologies of IPFS greatly reduce the cost of storage and backup and ensure the permanent storage of data.

8.3 System expansion analysis

The system adopts a split structure design in the working mode design, with data storage and data service implemented in IPFS distributed storage and FIBOS blockchain network respectively. Therefore, it has good scalability in

Fig. 14 Download transactions

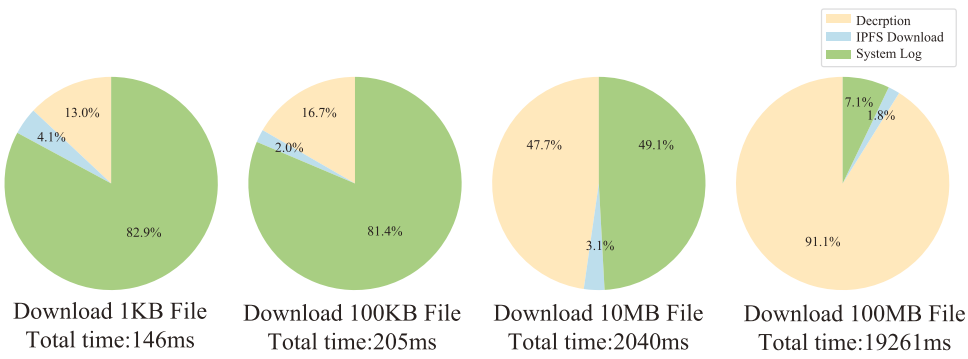


Table 3 Compare with Filecoin and Storj

Platforms	FIBPRO	Filecoin	Storj
Consensus	DPoS	PoS/PoW	Kademlia DHT
Blockchain Platform	FIBOS	Ethereum	Non-mainstream blockchain
Encryption and Privacy	AES/ECC	None	AES
Data Sharing	based on ECC	CID share	None
Data Auditing	Yes	Yes	Yes
Behavioral Auditing	Yes	None	Few
Decentralization	Very High	Very High	High
Availability	High	High	Very High
Stability	Very High	Moderate	High

data storage, and the storage capacity of the system can be directly expanded by IPFS node capacity increase. Since the data indexes occupy minimal space in the blockchain, the expansion in terms of storage capacity is close to infinite.

In addition to storage expansion, the system is designed with good scalability of data business functions. Unlike information sharing achieved directly with asymmetric secret keys, using a hybrid encryption system can effectively reduce redundant backups of the same data in the sharing process, which will significantly reduce storage overhead. At the same time, the sharing of uploaded data is not one-time, and through the information index recorded by blockchain, the sharer can realize multiple sharing of data. Notably, subsequent sharing actions can be initiated by the recipient in the previous sharing, which will enable fast sharing of data, and the system still retains only one copy of the encrypted ontology of the data during the whole process.

8.4 Performance comparison

As the current leading examples of commercially integrated distributed storage and blockchain-based services, Filecoin and Storj, we conducted a comparative analysis of FIBPRO in terms of its design and implementation as shown in Table 3. Furthermore, we assessed the data upload and download performance of all three platforms using cloud servers located in Hong Kong.

FIBPRO along with Filecoin adopts a structure that combines blockchain and IPFS. However, it differs in that Filecoin [37, 38] does not integrate any data encryption storage mechanisms and does not provide data behavior auditing. Storj [39, 40], on the other hand, defaults to using data encryption methods. Unlike both, it draws inspiration from blockchain concepts but does not fully integrate with blockchain in the conventional sense. Notably, its performance is outstanding, approaching or even surpassing that of cloud services.

Both Filecoin and Storj experienced operational failures and instability in their cloud server tests. This is related to their public chain-style design, with its sparse number of Asian storage nodes leading to a lower success rate. In the performance test, Filecoin had an average upload speed of 6.16MB/s and a download speed of 3.24MB/s, while Storj had an average upload speed of 14.13MB/s and a download speed of 8.79MB/s. FIBPRO obtained an average of 2.31MB/s for uploads and 5.29MB/s for downloads and encountered no operational failures during the test. Although we cannot confirm the actual hardware configurations of Filecoin and Storj for storage and data processing, our test environment for FIBPRO yielded performance averages that closely approached theirs. Given the additional overhead from encryption methods.

9 Conclusion

This paper introduced FIBPRO, a secure data storage and sharing system based on the consortium blockchain FIBOS. It is primarily designed to ensure the security and efficiency of Email-style information sharing through hybrid encryption and digital signature technology. Under the premise of guaranteed data security, FIBPRO proposes an efficient data-sharing scheme based on smart contracts. With optimized information-sharing transactions, information can be shared at a much smaller cost, while having the ability to be shared multiple times. In addition, FIBPRO includes a set of custom blockchain auditing modules that provide more efficient transaction filtering and local data backup. In the process of FIBPRO experimental analysis, a quantitative evaluation of the blocking efficiency of the blockchain and the speed of data upload, sharing, and download was carried out. The results showed that FIBPRO was reasonable and effective, which precisely met the basic design expectations.

In future work, we intend to optimize on-chain contract execution and off-chain co-working logic to reduce blockchain contract execution overhead and improve system response efficiency. At the same time, we are considering a combination of searchable encryption technology and custom auditing to further expand FIBPRO's system disaster recovery and backup capabilities.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No.61906099), the Open Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources (No.KF-2019-04-065).

Author contributions Rui Han: Conceptualization, Methodology, Writing Original Draft. Wang Yu: Software, Data curation, Analysis. Mingfa Wan: Proofread, Modelling, Writing - Review & Editing. Yuan Teng: Software, Implementation. Guozi Sun: Methodology, Supervision.

Funding This work was supported by the National Natural Science Foundation of China (No.61906099), the Open Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources (No.KF-2019-04-065).

Data availability The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

Declarations

Ethics approval Not applicable.

Consent for publication All authors unanimously agree to publish the paper.

Competing interests The authors declare no competing interests.

References

1. Nachiappan R, Javadi B, Calheiros RN et al (2017) Cloud storage reliability for big data applications: a state of the art survey. J Netw Comput Appl 97:35–47
2. Zhang Y, Geng H, Su L et al (2022) A blockchain-based efficient data integrity verification scheme in multi-cloud storage. IEEE Access 10:105920–105929
3. Prajapati P, Shah P (2022) A review on secure data deduplication: Cloud storage security issue. J King Saud Univ Comput Inf Sci 34(7):3996–4007
4. Nakamoto S, Bitcoin A (2008) A peer-to-peer electronic cash system. Bitcoin. https://bitcoin.org/bitcoin-pdf-4:2
5. Yaga D, Mell P, Roby N et al (2019) Blockchain technology overview. arXiv preprint arXiv:1906.11078
6. Sanka AI, Cheung RC (2021) A systematic review of blockchain scalability: Issues, solutions, analysis and future research. J Netw Comput Appl 195:103232
7. Yu K, Tan L, Aloqaily M et al (2021) Blockchain-enhanced data sharing with traceable and direct revocation in iiot. IEEE Trans Industr Inf 17(11):7669–7678
8. Arooj A, Farooq MS, Umer T (2022) Unfolding the blockchain era: Timeline, evolution, types and real-world applications. J Netw Comput Appl 103511
9. Fu JS, Liu Y, Chao HC et al (2018) Secure data storage and searching for industrial iot by integrating fog computing and cloud computing. IEEE Trans Industr Inf 14(10):4519–4528
10. Li J, Wu J, Chen L (2018) Block-secure: Blockchain based scheme for secure p2p cloud storage. Inf Sci 465:219–231
11. Chen Y, Ding S, Xu Z et al (2019) Blockchain-based medical records secure storage and medical service framework. J Med Syst 43(1):1–9
12. Mustafa M, Alshare M, Bhargava D et al (2022) Perceived security risk based on moderating factors for blockchain technology applications in cloud storage to achieve secure healthcare systems. Comput Math Methods Med 2022
13. Nizamuddin N, Abugabah A (2021) Blockchain for automotive: an insight towards the ipfs blockchain-based auto insurance sector. Int J Electr Comput Eng (IJECE) 11
14. Khatal S, Rane J, Patel D et al (2021) Fileshare: a blockchain and ipfs framework for secure file sharing and data provenance. In: Advances in Machine Learning and Computational Intelligence. Springer, pp 825–833
15. Kumar S, Bharti AK, Amin R (2021) Decentralized secure storage of medical records using blockchain and ipfs: a comparative analysis with future directions. Secur Privacy 4(5):e162
16. Wang S, Wang H, Li J et al (2020) A fast cp-abe system for cyber-physical security and privacy in mobile healthcare network. IEEE Trans Ind Appl 56(4):4467–4477
17. Sun X, Yu FR, Zhang P et al (2021) A survey on zero-knowledge proof in blockchain. IEEE Netw 35(4):198–205
18. Ta AT, Khuc TX, Nguyen TN et al (2021) Efficient unique ring signature for blockchain privacy protection. In: Information Security and Privacy: 26th Australasian Conference, ACISP 2021, Virtual Event, December 1–3, 2021, Proceedings 26, Springer, pp 391–407
19. Cao Y, Li Y, Sun Y et al (2019) Decentralized group signature scheme based on blockchain. 2019 International Conference on Communications. Information System and Computer Engineering (CISCE), IEEE, pp 566–569
20. Čapko D, Vukmirović S, Nedić N (2022) State of the art of zero-knowledge proofs in blockchain. In: 2022 30th Telecommunications Forum (TELFOR), IEEE, pp 1–4
21. Mallaki M, Majidi B, Peyvandi A et al (2021) Off-chain management and state-tracking of smart programs on blockchain for secure and efficient decentralized computation. Int J Comput Appl 1–8
22. Bamakan SMH, Motavali A, Bondarti AB (2020) A survey of blockchain consensus algorithms performance evaluation criteria. Expert Syst Appl 154:113385
23. Lepore C, Ceria M, Visconti A et al (2020) A survey on blockchain consensus with a performance comparison of pow, pos and pure pos. Mathematics 8(10):1782
24. Saleh F (2021) Blockchain without waste: Proof-of-stake. Rev Financial Stud 34(3):1156–1190
25. Luo Y, Chen Y, Chen Q et al (2018) A new election algorithm for dpos consensus mechanism in blockchain. In: 2018 7th international conference on digital home (ICDH), IEEE, pp 116–120
26. Fan X, Chai Q (2018) Roll-dpos: a randomized delegated proof of stake scheme for scalable blockchain-based internet of things systems. In: Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, pp 482–484
27. Zou W, Lo D, Kochhar PS et al (2019) Smart contract development: Challenges and opportunities. IEEE Trans Softw Eng 47(10):2084–2106
28. Webassembly (2023) https://webassembly.org/. Accessed 16 Mar 2023
29. Mohan AP, Gladston A et al (2020) Merkle tree and blockchain-based cloud data auditing. Int J Cloud Appl Comput (IJCAC) 10(3):54–66
30. Kang P, Yang W, Zheng J (2022) Blockchain private file storage-sharing method based on ipfs. Sensors 22(14):5100
31. Zheng Q, Li Y, Chen P et al (2018) An innovative ipfs-based storage model for blockchain. In: 2018 IEEE/WIC/ACM international conference on web intelligence (WI), IEEE, pp 704–708
32. Ipfs (2023) https://docs.ipfs.tech/. Accessed 14 Aug 2023
33. Grant G, Hogan R (2015) Bitcoin: Risks and controls. J Corp Account Finance 26(5):29–35
34. Liu H, Liu C, Zhao W et al (2018) S-gram: towards semantic-aware security auditing for ethereum smart contracts. In: 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE), IEEE, pp 814–819
35. Sheldon MD (2019) A primer for information technology general control considerations on a private and permissioned blockchain audit. Curr Issues Audit 13(1):A15–A29
36. Bonyuet D (2020) Overview and impact of blockchain on auditing. Int J Digit Account Res 20:31–43
37. Filecoin (2023) https://docs.filecoin.io/. Accessed 16 Aug 2023
38. Guidi B, Michienzi A, Ricci L (2022) Evaluating the decentralisation of filecoin. In: Proceedings of the 3rd International Workshop on Distributed Infrastructure for the Common Good, pp 13–18

39. Storj v3 (2023) Whitepaper. https://www.storj.io/whitepaper. Accessed 16 Aug 2023
40. Li H, Mi X, Dou Y et al (2023) An empirical study of storj dcs: Ecosystem, performance, and security. In: 2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS), IEEE, pp 1–10

Rui Han received the B.Sc degree from Nanjing University of Posts and Telecommunications, China. He is currently working toward the M.Sc degree in the School of Computer Science and Technology at Nanjing University of Posts and Telecommunications. China. His research interests include blockchain application security and distributed application development.

Yu Wang was born in Nantong Jiangsu Province China. He is pursuing a master's degree at Nanjing University of Posts and Telecommunications. His research includes blockchain-enabled searchable encryption, the privacy-preserving framework for blockchain networks.

Mingfa Wan was born in Guangxi, China. Studying for a master's degree at Nanjing University of Posts and Telecommunications, his main research includes information privacy protection, blockchain development, and application.

Teng Yuan was born in Jiangsu Province, China. He is pursuing a master's degree at Nanjing University of Posts and Telecommunications. His research includes web, blockchain, and DeFi.

Guozi Sun is a professor in the School of Computer Science and Technology at Nanjing University of Posts and Telecommunications, China. His research interests include blockchain forensics, digital forensics, and digital investigation. Sun received his Ph.D. in mechanical engineering and automation from Nanjing University of Aeronautics and Astronautics, China. He is a member of the IEEE Computer Society, ACM, China Computer Federation (CCF), Chinese Institute of Electronics (CIE), and Information Security and Forensics Society (ISFS), China.