Future Generation Computer Systems 141 (2023) 197–204

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A peer-to-peer file storage and sharing system based on consortium blockchain

Shaoliang Peng [a,b,*], Wenxuan Bao [a], Hao Liu [a], Xia Xiao [a], Jiandong Shang [c], Lin Han [c], Shan Wang [d,e], Xiaolan Xie [f,*], Yang Xu [a,*]

[a] College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[b] The State Key Laboratory of Chemo/Biosensing and Chemometrics, Hunan University, Changsha 410082, China
[c] National Supercomputing Center in Zhengzhou, Zhengzhou University, Zhengzhou 450001, China
[d] Faculty of Arts and Humanities, University of Macau, Macau 999078, Macao Special Administrative Region of China
[e] Institute of Collaborative Innovation, University of Macau, Macau 999078, Macao Special Administrative Region of China
[f] College of Information Science and Engineering, Guilin University of Technology, Guilin 541004, China

ARTICLE INFO

Article history:
Received 21 February 2022
Received in revised form 5 September 2022
Accepted 10 November 2022
Available online 13 November 2022

Keywords:
Peer-to-peer file system
Data sharing
Consortium blockchain

ABSTRACT

In the era of big data, data is playing an increasingly important role in scientific study, and reliable storage and secure sharing of data have become a research hotspot. At present, centralized solutions based on data centers and cloud storage have problems with data-right confirmation and center trust. A large number of decentralized storage solutions are public systems, in which blockchain technology, as a tool for value exchange, does not solve the problems of data verification and system supervision. We propose a peer-to-peer storage system with identity access, which achieves data validation, cross-organizational data retrieval, trusted authorization, and sharing based on the consortium blockchain. Our solution proposes a peer-to-peer data storage scheme based on the consortium blockchain and a set of identity authentication mechanisms compatible with the consortium blockchain. Based on this, we propose a blockchain-based permission control scheme and a set of retrieval, authorization, and sharing processes. Finally, we implemented and tested the system to prove the feasibility of the scheme.

1. Introduction

With the advent of the era of big data, a large number of data-driven research paradigms have emerged, in which people can uncover some valuable information from a large amount of seemingly disorganized data and use it to guide people to make more rational decisions [1]. Data plays an increasingly critical role in scientific research, and how to store data reliably and share it securely is an important prerequisite for its proper utilization.

Reliable data storage has long been a topic of concern. Usually, an organization will establish its own data center [2] to ensure the reliability of data storage and use data backup and recovery technology to prevent data loss. At present, cloud storage technology has attracted widespread attention since it is safe, reliable, and easy to use. Many enterprises store data on cloud storage platforms. Data sharing can promote the discovery of more valuable information from data. Ref. [3] has demonstrated that the scale of the training set will impact the precision of machine learning. However, there were frequently insufficient samples available for the experiment. Accordingly, researchers can collect wide and diverse data to provide more accurate and objective study results if data may be used across enterprises.

Data platforms were typically used to realize data sharing before distributed ledger technology emerged. Large organizations will construct organization-level data warehouses or big data platforms to enable consolidated internal data collecting, processing, storage, and application. This is especially true for government agencies and sizable financial institutions. However, as no parties are prepared to actively share data, there are relatively few instances of data sharing through the development of a single, centralized data platform among numerous organizations, particularly those with peer-to-peer interactions.

By introducing smart contracts, identity verification and privacy protection mechanisms, blockchain data can be shared among users [4]. The decentralized feature of blockchain also guarantees the security of data. All nodes record the ledger data, thus preventing the data from being maliciously tampered with. When data is accidentally lost, the information can be reacquired through other nodes. Distributed ledgers record transactions between users, and this shared ledger also reduces the

[*] Corresponding authors.
E-mail addresses: slpeng@hnu.edu.cn (S. Peng), xie_xiao_lan@foxmail.com (X. Xie), xuyangcs@hnu.edu.cn (Y. Xu).

https://doi.org/10.1016/j.future.2022.11.010

time and confirmation costs incurred by mediating different ledgers. However, the redundant data also creates load pressure on the nodes [5]. A blockchain system's accessibility effectively disappears once it has recorded a certain volume of data.

The usual solution to the above problem is that the file remains stored on the file system, while the blockchain only stores the fingerprint and other metadata information of the file [6]. Blockchain is a decentralized tool, and the application scenarios are also among decentralized organizations. The file system for storing the file is obviously better suited to adopt a decentralized file system in line with the blockchain when the blockchain and the storage file are separated.

At present, blockchain systems can be mainly classified into public blockchain and consortium blockchain according to whether there is access control. Distributed file systems based on public blockchain are usually without access control. There are many such systems, such as IPFS, Swarm, Storj, and so on [7–9]. These systems are essentially decentralized cloud storage systems. Their combination with blockchain mainly provides a business operation model for the system, where one party sells storage resources to get tokens and the other party spends the tokens to buy storage resources. On the one hand, the public blockchain used by these systems is difficult to supervise. On the other hand, the value of data is often much greater than the value of the physical devices that store it. This operating model in which the system primarily considers selling storage space is completely missing the point. This paper introduces a peer-to-peer file storage and sharing system relying on the consortium blockchain in consideration of the drawbacks of current data sharing storage strategies. The system adopts a verification method applicable to the consortium blockchain for identity authentication. Compared to public systems, this scheme only allows specific member storage systems to improve system security and make the system manageable. A fault tolerance mechanism based on erasure code is introduced into the storage system to avoid single point of failure. The system also proposes a cross-organizational file retrieval and authorization mechanism for data sharing.

The rest of the paper is organized as follows. Section 2 introduces the work related to our system, including the centralized data storage and sharing solutions and the decentralized blockchain-based cloud storage system. Section 3 illustrates the specific design of our system, including the system architecture, the authentication mechanism and access control, peer-to-peer file storage, and file retrieval and authorization. Section 4 presents the system implementation details and evaluates its performance. Finally, Section 5 summarizes our work.

2. Related work

2.1. Centralized data storage and sharing scheme

Centralized storage and sharing solutions can be divided into two main types, which are data center-based solutions and cloud-based solutions.

Data center-based solutions are usually built on the original data center of each organization. Each organization may build its own data center in different periods and store its own data in its own data center. When they have data sharing needs, the general process is shown in Fig. 1. A typical application scenario of this data sharing scheme is data sharing among PACS systems in hospitals. This kind of system can realize the sharing of imaging data among clinical departments and help doctors make a more accurate diagnosis [10], but it is difficult to be applied to the storage and access of large-scale data. Although such systems achieve data sharing based on the original data center, they have many security problems. First of all, such systems do not have a good mechanism to identify data rights, which can easily result in disputes on data ownership. Secondly, the data authorization record of such systems is unilateral, and there is a security risk of being denied by the authorized person.

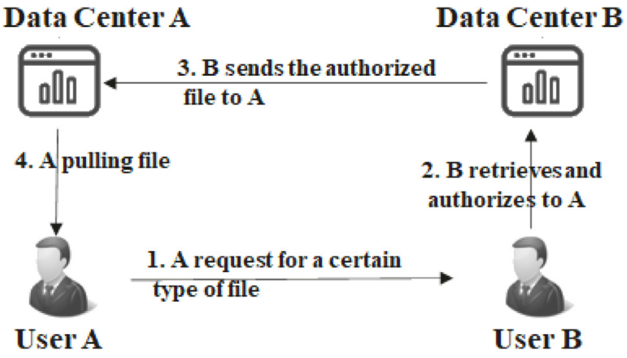

Data Center A
Data Center B
3. B sends the authorized file to A
4. A pulling file
2. B retrieves and authorizes to A
1. A request for a certain type of file
User A
User B

Fig. 1. General process of data sharing between data centers.

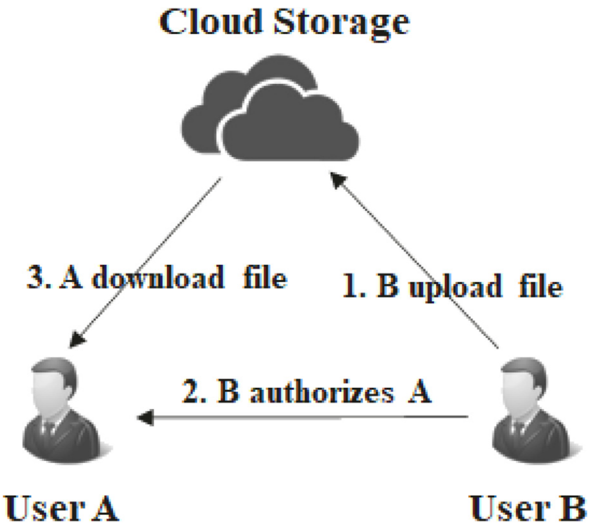


Fig. 2. General process of cloud storage data sharing.

In recent years, with the rise of cloud storage technology, data sharing scheme based on cloud storage has also become a hotspot of research and application. In this scheme, both data owners and data users use the same third-party cloud storage service, and its general process is shown in Fig. 2. There are many security risks in simple cloud storage, such as misconfiguration, poor access control and lack of data governance. In recent years, data leakage events of cloud storage occur frequently. Many researchers begin to pay attention to the security problems of cloud storage data sharing. Ref. [11] proposes a data sharing scheme based on cloud storage, which uses the secure interactive proof system to solve the security problems of data sharing in cloud storage. Ref. [12] improved revocable and identity-based conditional proxy re-encryption scheme with ciphertext evolution for secure and efficient cloud data sharing. However, cloud storage based solutions are still centralized. Cloud storage service providers unilaterally determine the availability of the system. Data migration issues will arise if cloud storage companies stop offering their services. Also, the rights of data owners are difficult to preserve since such systems lack a reliable data-right confirmation mechanism. This also leads to cloud storage data being vulnerable to illegal access and tampering by attackers.

2.2. Blockchain-based storage system

Through a decentralized network, blockchain system assures the safety and integrity of data storage and exchange across a wide range of applications. In respect of blockchain-enabled technologies, Taylor et al. [13] found that data sharing and storing rank second only to IoT technology in terms of popularity. This research covers many fields, including tamper-proof mechanisms, encrypted storage, and blockchain systems. Building reliable blockchain systems for data sharing is a recent research hotspot.

There are restrictions on how blockchain data is stored. According to Zheng et al. [14], the blockchain adopts a decentralized mechanism, and the ledger data is saved at all nodes at the same time. In this way, the data is almost never lost, and the nodes can restrict each other to prevent malicious tampering. However, it will generate a large amount of redundant information and seriously reduce the efficiency of the transaction.

Searching for solutions to decrease blockchain redundancies is one option to improve the performance of storage. A distributed storage method utilizing encoding was published by Dai et al. [5] . The fundamental strategy is constructing blocks and separating them into smaller units. The system then transmits the additional sub-blocks, which are generated from the original sub-blocks, to all other nodes. This method decreases storage space and makes the already underperforming blockchain system more inefficient since it produces a mass of block requests on the network while searching for blocks. A strategy proposed by Jia et al. [15] can adjust the data repetition rate on the blockchain. Each full node does not store all the ledger information but dynamically adjusts and allocates the data it needs to store according to the algorithm. This mechanism reduces the redundancy of data but also reduces the recoverability of files. The most common method for addressing blockchain data storage issues is off-chain storage . The study by Huang et al. [4] applied two popular off-chain storage technologies, Swarm and IPFS, and evaluated the architecture, which is composed of a large number of layers. Rather than a standard blockchain in which each accounting node stores all of the info, the particular blockchain is loosely coupled with the data storage module and only interlinked via a network port. This scheme significantly reduces the load on the system and increases its efficiency. The majority of the existing public blockchain storage solutions use blockchain-specific tokens as a measure of value for the resource . For instance, a public blockchain coin called FileCoin is utilized by IPFS to pay for resources [7]. Swarm is an off-chain storage scheme [8]. As a solution encouraged by Ethereum, Storj issues its own token exclusively for exchanging storage resources. At the same time, Storj also allows other tokens such as Ethereum to be used as exchange credentials [9]. Few solutions adopt consortia blockchain-based distributed storage systems to address the issue of data verification and permission. We build a blockchain-based peer-to-peer file system, which is a decentralized system, as shown in Fig. 3. Users will actively store their data encrypted into this storage system, and data demanders can actively retrieve their interested data from the system, and then use that data after authorization from the data owner. The system extends the application scope of the system based on MIFS [16], and carries out full-text retrieval based on the file name, so as to support the retrieval of a wider range of file types and provide a more general scheme for decentralized file sharing.

3. Specific design

3.1. System architecture

Our system is a peer-to-peer structured storage system to solve the problem of file sharing among organizations. Fig. 4 illustrates the system architecture.

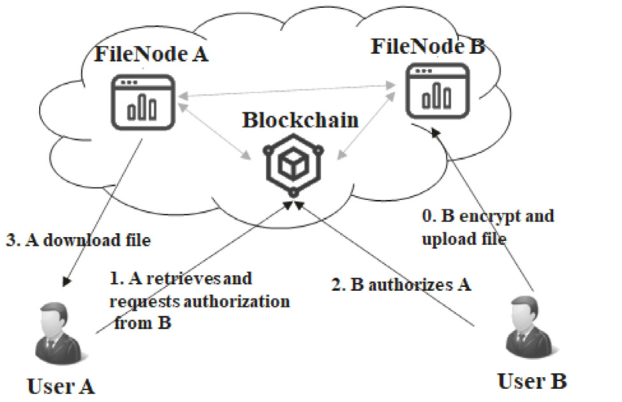


Fig. 3. General process of data sharing in our system.

Organizations must work together to create a consortium blockchain and a peer-to-peer file storage system based on it, where the storage system is made up of the peer file nodes (FileNodes) of each organization. The following functions are required for each FileNode: a blockchain data interface, file storage, identity authentication, access control, and file retrieval. The details of each item are shown below.

- Blockchain Data Interface: Used to supply the system with consortium blockchain verified data access services. The consortium blockchain provides a smart contract, including a complete set of transactional operations for modifying file metadata. Members of the consortium can query and edit the file metadata through the blockchain data interface, which offers data dependencies for storage issues and access control.
- File Storage: Used to store files uploaded by organization users. Members' submitted files are divided into slices and then kept in separate nodes. It requires identification authentication and access permission to grant other organizations access to the file slices that are kept by the organization.
- Identity Authentication: Used to enable identity access to storage nodes. The system only accepts nodes that have passed identity authentication. It is equipped with an identity issuer and an authenticator. The identity issuer is able to issue identity certificates to nodes and users, and the authenticator is able to verify the correctness of the certificates issued by the identity issuer.
- Access Control: Used to confirm whether an organization has authorization to access a file. This module is realized by the permission grantor and the permission verifier. Access to his organization's files must be granted by the permission grantor to other organizations. The file slice requester must have permission to access the file in question, according to the permission verifier. The request's authorization to access this slice is confirmed by the permission verifier.
- File Retrieval: Used to retrieve files of interest from the system. Considering the diversity of files in the system, the system adopts the full-text retrieval strategy based on file name rather than the retrieval strategy based on file content. The implementation of this function needs to establish an inverted index in FileNode based on file name, and users can search the required files based on keywords.

FileNode relies on the metadata that is kept on the blockchain. The storage system's data structure is depicted in Fig. 5.

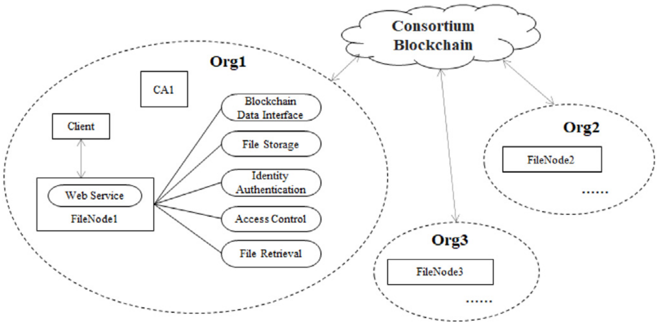


Fig. 4. General architecture.

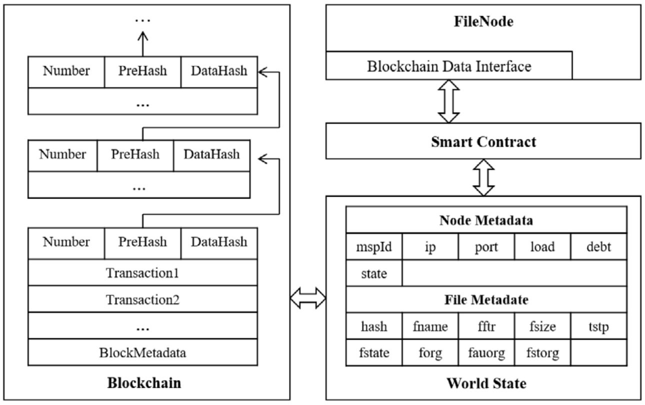


Fig. 5. The data structure of consortium blockchain.

The system includes two different forms of metadata: node metadata and file metadata, which provide indicative information for the system at the level of storage medium and data, respectively. Node metadata is the status data for each FileNode, including node id, node address, etc. And the file metadata includes file hash, file name, etc. and slice storage organization. Smart contracts are used to edit this metadata, and the records of the operations are bundled into blocks and saved in the blockchain. The smart contract mainly provides the following transaction operations.

- InsertNode: Insert a node metadata. This transaction guarantees that the inserted node must belong to the organization that submitted this transaction.
- SelectAllNode: Get all node metadata. No permission control for this transaction.
- InsertFile: Insert a file metadata. This transaction ensures that the organization to which the file belongs is the same as the organization that submitted this transaction.
- SelectFile: Get file metadata by file hash. No permission control for this transaction.
- SelectAllFile: Get all file metadata. No permission control for this transaction.
- DeleteFile: Update a file's metadata to a deleted state. This transaction ensures that the organization to which the file belongs is the same as the organization that submitted this transaction.
- AddAuthorizedOrganization: Update a file's metadata with the users who have access to the file. This transaction ensures that the organization that submitted it is the one that owns the file.

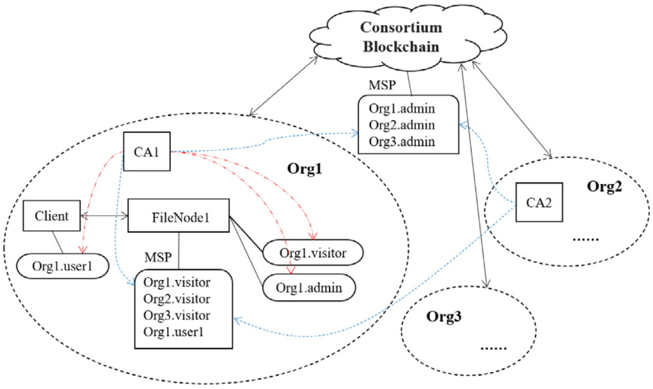


Fig. 6. The authentication mechanism.

3.2. *Authentication and access control*

The system's key characteristic that sets it apart from other public blockchain systems is the authentication mechanism, which is a crucial prerequisite for system implementation. The consortium blockchain's unique properties dictate that they have a separate authentication system. With the authentication mechanism suggested in this work, FileNodes and consortium blockchain can share a compatible authentication mechanism.

Fig. 6 depicts the authentication system, including a certificate authority (CA) and a member identity provider (MSP).

An organization is a group of nodes that have the same root CA or intermediate CA, and the MSP defines which certificates can be signed by the root CA or intermediate CA to be accepted. A consortium blockchain network can be managed by one or more MSPs, and each organization has to have its own MSPid when setting up a consortium blockchain.

The CA issues encrypted verified digital certificates to clients or nodes, signs these certificates, and binds the public keys of participants to them. Depending on the identity, these certificates can be classified into three types: user, administrator, and visitor. MSP is a set of configuration folders that are added to the network, which contains the list of licensed identities. It manages and verifies the identities of members so that they are granted different privileges. Clients can only obtain a user identity after which they can access FileNode and make service requests to it. FileNodes get visitor identity or administrator identity through CA. The former can access FileNodes of other organizations, and the latter can manage the consortium blockchain.

Authentication is a prerequisite for gaining access to nodes, which provides data security for the system. Before executing following actions, users or nodes need to pass through access control, and the system checks the visitor's identification to determine whether they are authorized to do so. The blockchain contains the access control dependency data that allows it to define and manage user identities and access rights. Access control rules are then used to verify permissions. The guidelines for access control are listed below.

- File Upload/Download: Only user with a User identity belongs to the same organization as the FileNode can upload/download files.
- File Deletion: Users of their own organization can delete any file on their organization's FileNode.
- Slice Upload: Any legitimate node with a Visitor identity can upload a slice. FileNode needs to verify that the file is recorded on the consortium blockchain and that the user to whom the file belongs and the user who uploads the file

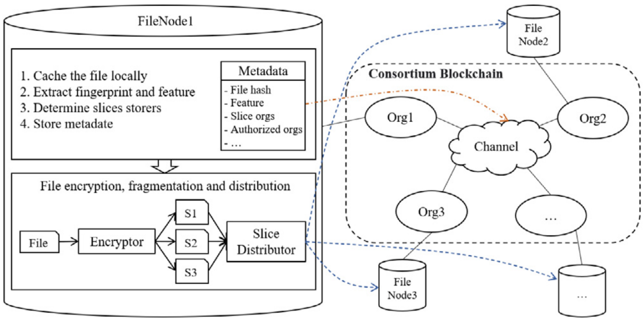


Fig. 7. File peer-to-peer stored procedures.

slice should be the same. This ensures that the file slice is uploaded by the user to whom the file belongs, so FileNode does not save fake file slice.

- Slice Download: Only user with a Visitor identity from the file owner's organization and from the authorized organization can download file slice.
- Slice Deletion: Only user with a Visitor identity from the file owner's organization can delete a file slice.

3.3. Peer-to-peer file storage

During the storage process, the system processes the files into encrypted slices and distributes them to its selected nodes. The precise processing sequence is depicted in Fig. 7.

Step 1, the system temporarily stores the file uploaded by the user.

Step 2, obtain the basic information about the file and calculate its hash value based on its features.

Step 3, the system selects FileNodes for file slices according to the difference between the debt and load of the node.

Step 4, store the metadata of the file in the blockchain.

Step 5, process the original file into encrypted file slices and transfer them to the previously selected FileNodes.

The load and debt of a node each indicate the file sizes that are stored on that node and are distributed by the organization to other nodes, respectively. We demand that the load and debt of a node be in balance in order to guarantee equitable storage among organizations in the system. As a result, while choosing the slice storers, preference is given to the node with the largest difference between the values of debt and load.

In the fifth step, the system will generate a random key and use it to encrypt the original file. The key is stored in the FileNode and only the provider of the file has access to it. Next, the file is encoded into three file slices. To enhance the security of the data, the system adopts a backup technique, and 1/3 slices is redundant. Depending on Reed Solomon Codes [17], the system's encoding technique, we can rely on any of these slices to retrieve the original file, allowing the system as a whole to endure a temporary FileNode outage.

Note that the order of the fourth step and the fifth step cannot be interchanged, because FileNode will check the source of the file slice before storing it. The FileNode will not keep the file not required by the storage system. When FileNode encounters a request to store a file slice, it will not only determine if the file slice of the blockchain file should be saved locally, but also verify that the sender of the file slice is the file owner. The file slice will only be stored if both requirements are satisfied. The major goal of this technique is to stop hostile nodes from performing garbage file write attacks, in which a lot of files that are not part of the system are written, making FileNodes unavailable. This strategy guarantees the security of the system.

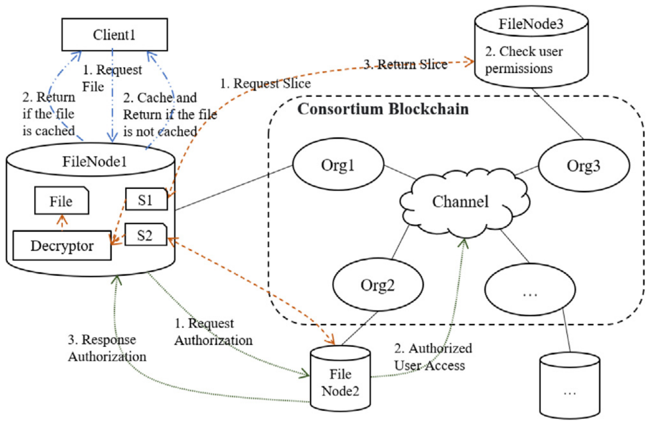


Fig. 8. Authorization and sharing process.

3.4. Retrieval, authorization, and sharing

On purpose of promoting data sharing, our system realizes the function of file retrieval. Considering the diversity of file types in the system, the system uses a full-text retrieval strategy based on file name.

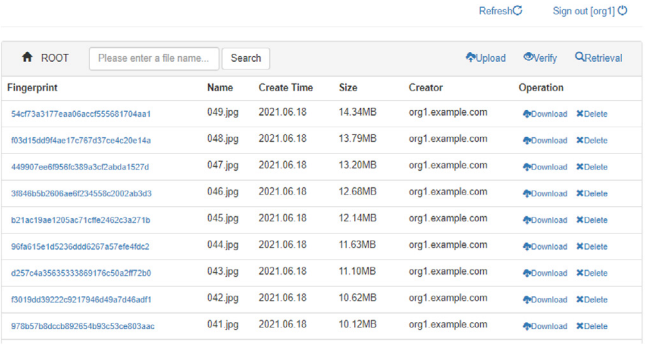
FileNode needs to maintain an inverted index locally in order to realize the global retrieval function. When the blockchain writes a piece of file metadata, FileNode will listen to the writing operation of the blockchain and obtain the file name and file hash. Then, FileNode establishes an inverted index based on the file name locally, while the index points to the file hash. In this way, the FileNode of each organization actually maintains the full-text index information of all files in the system, so that cross-organizational file retrieval can be realized.

When a user has a file retrieval requirement, it first sends a retrieval keyword to the FileNode. The FileNode queries the local inverted index, obtains the group of files that best meet this keyword, and returns the metadata of this group of files to the user. We will add a shared lock to the index before reading and an exclusive lock to the index before writing because the inverted index is maintained by FileNode and there are concurrent read and write operations. In other words, a read operation will be blocked while the index is being updated until the update is finished.

The process of users requesting authorization and sharing files is shown in Fig. 8.

The download permission is only available to vetted users. The authorization request is initiated by the data demander. If the data provider allows this request, it changes the file's attributes to accessible and delivers the corresponding symmetric key to the former.

The user obtains the authorized file according to the following steps. The user first sends a message to the Filenode requesting to obtain the target file; the FileNode then determines whether the target file is already cached internally, feedbacks the situation to the user, and starts the caching process if the file is not found internally. During the caching process, FileNode will traverse each organization's FileNodes until the slices of the target file are all found. Next, it will get the file slices and put them together into the original encrypted file. Finally, it will decrypt the original file with the symmetric key it received earlier and transmit the processed file to the user.



Fingerprint	Name	Create Time	Size	Creator	Operation
54cf73a3177eaa06accf555681704aa1	049.jpg	2021.06.18	14.34MB	org1.example.com	Download Delete
f03d15dd9f4ae17c767d37ce4c20e14a	048.jpg	2021.06.18	13.79MB	org1.example.com	Download Delete
449907ee6f956fc389a3cf2abda1527d	047.jpg	2021.06.18	13.20MB	org1.example.com	Download Delete
3f846b5b2606ae6f234558c2002ab3d3	046.jpg	2021.06.18	12.68MB	org1.example.com	Download Delete
b21ac19ae1205ac71cffe2462c3a271b	045.jpg	2021.06.18	12.14MB	org1.example.com	Download Delete
96fa615e1d5236ddd6267a57efe4fdc2	044.jpg	2021.06.18	11.63MB	org1.example.com	Download Delete
d257c4a35635333869176c50a2ff72b0	043.jpg	2021.06.18	11.10MB	org1.example.com	Download Delete
f3019dd39222c9217946d49a7d46adf1	042.jpg	2021.06.18	10.62MB	org1.example.com	Download Delete
978b57b8dccb892654b93c53ce803aac	041.jpg	2021.06.18	10.12MB	org1.example.com	Download Delete

Fig. 9. User interface for peer file node.

Table 1
The nodes to be deployed for each hospital.

Node	Quantity	Function
CA	1	Issuing User, Admin, and Visitor identity to the organization's nodes and users
Peer	1+	Used in Fabric to store blockchain data and to endorse the results of smart contract execution
Orderer	1+	Used in Fabric to order transactions and make transaction data consistent across nodes
Filenode	1	Used to form a peer-to-peer file storage system

4. Implementation and evaluation

4.1. System implementation

Hyperledger Fabric is one of the most popular and adopted blockchain frameworks, allowing participating organizations to join and communicate with each other, provided that members are registered from a trusted MSP. It is used to implement the consortium blockchain network on which the system depends because its identity authentication process is compatible with that of the peer-to-peer file system. So no additional steps are required for the organizations to authenticate.

We used Java to establish peer-to-peer file nodes on the blockchain. Each node is interconnected with one another using the RESTFUL API, a layered architecture, to realize information transmission. Peer file nodes offer RESTFUL interface functions to other peer file nodes in the network and offer web services to the authorized users.

The User identity can be shown as illustrated in Fig. 9 after logging in. Organization users can access the peer file node of the organization using this User identity. Through this web application, users can control the peer file node to store, retrieve, authorize, and share files within the system.

4.2. Performance

To validate the performance of the system, we established a consortium blockchain comprising three test organizations. The deployment of nodes in each organization is present in Table 1.

In our experiments, all nodes were deployed in docker containers. The machine running these containers had Centos 6 OS and CPU of Intel(R) Xeon(R) CPU E7-8890 v3 @ 2.50 GHz and a 36 GB memory.

File Write Performance: To get a comprehensive evaluation of the system's write performance, we measured its write speed for files of different sizes. In the experiments, the network transfer latency of the files is disregarded in order to eliminate irrelevant variables, and only the time consumption between the user finishing the upload and the server returning a response is recorded.

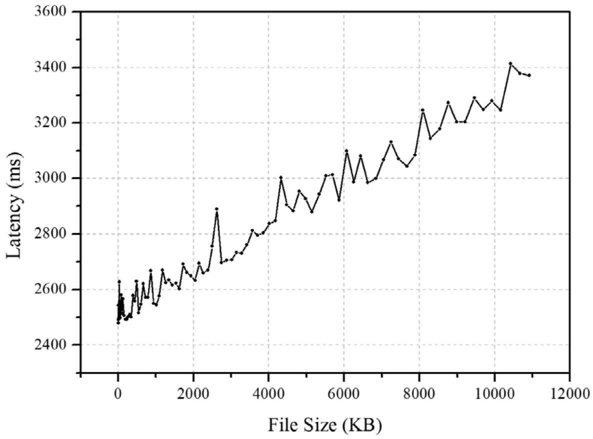


Fig. 10. Impact of upload file size on server processing latency.

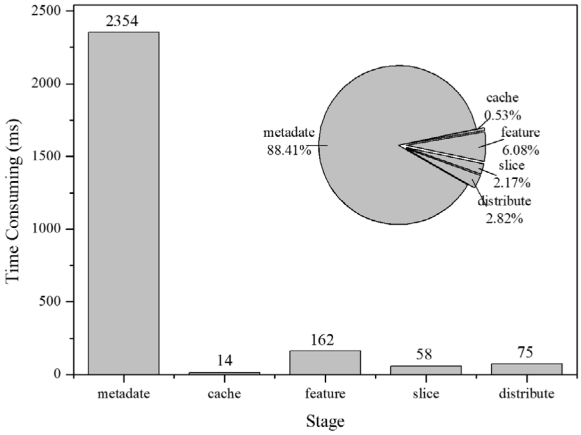


Fig. 11. Time consumption of the server for each stage of processing the uploaded file.

For each size, we prepared ten files to rule out chance in the results. The jmeter script is used in the experiment to upload these files. Fig. 10 depicts the outcomes. The figure shows a linear relationship between the file scale and processing time. As the file size increases, the latency also rises.

Next, we precisely measured the time spent on each step by the server to process the uploaded file. We produced 100 files of 1M size, which were sequentially transferred to the storage system via scripts. Depending on the file format, the upload process was divided into five steps: metadata, cache, feature, slice, and distribute, and the time consumption of each step was recorded separately. Finally, we calculated the average time consumption of each step based on the statistics and plotted the results in Fig. 11. As can be seen, the sending metadata phase consumes the heaviest proportion of time, at 88.41%, far more than the sum of the other phases. Therefore, the read and write speed of the blockchain is the key factor limiting the performance of the system, and the primary breakthrough for improving storage efficiency.

File Read Performance: To check the system's read latency, we upload multiple files of different sizes through a jmeter script. Similarly, the results exclude the delay in network transmissions

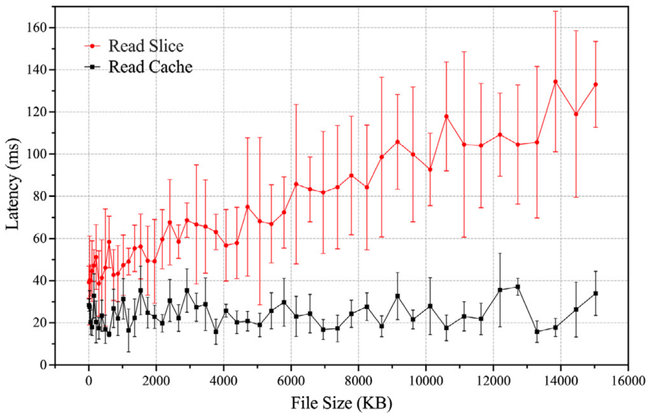


Fig. 12. Impact of file size on server processing latency when reading slice/cache.

and only represent the time the server takes to handle the download request. When the system accesses an authorized file for the first time, it must locate the file slices saved in the FileNodes, acquire all of them, and reassemble them as a whole. After that, the file is no longer read through the node but through the cache. Thus, "Read Slice" and "Read Cache" have distinct server processing latency, which need to be recorded separately. We can see the latency of these two reading approaches in Fig. 12. For each file size, we performed multiple sets of replicated experiments, and the figure includes the mean and standard deviation of the latency at different download sizes. From the data curves, we can see that there is essentially no relationship between the scale of the file and the latency of the server to read the cache. However, the file scale and the server processing latency are linearly correlated when reading the file slice. In the experiment, the average processing latency of the server for local caches of different sizes did not exceed 40 ms and varied relatively steadily compared to reading file slices.

5. Conclusion

This paper presents a system for reliable storage and secure sharing of data across organizations. The system uses a consortium blockchain for cross-organizational authentication and data access, and creates a peer-to-peer file system to store files. Files uploaded by organizations are processed into encrypted file slices and stored in the corresponding node.

The system introduces a consortium blockchain-compatible authentication mechanism for identity access and data validation. It adopts an identity-based access control scheme that grants different operation privileges for different visitors and introduces a fault tolerance mechanism based on erasure code to improve the robustness of the system.

Finally, this paper builds Fabric-based file nodes and tests the data reading and writing performance under different conditions. The test results prove the feasibility of this peer-to-peer file sharing system.

CRediT authorship contribution statement

Shaoliang Peng: Conceptualization. **Wenxuan Bao:** Methodology, Writing. **Hao Liu:** System implementation and experiment. **Xia Xiao:** Formal analysis. **Jiandong Shang:** Visualization. **Lin Han:** Investigation. **Shan Wang:** Correction. **Xiaolan Xie:** Project administration. **Yang Xu:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The link to the code is https://github.com/liuhaohn/haofs

Acknowledgment

This work was supported by NSFC, China Grants U19A2067, 61772543; Science Foundation for Distinguished Young Scholars of Hunan Province, China (2020JJ2009); National Key R&D Program of China 2017YFB0202602, 2018YFC0910405, 2017-YFC13 11003, 2016YFC1302500; Science Foundation of Changsha, China Z202069420652, kq2004010; JZ20195242029, JH2019-9142034; The Funds of State Key Laboratory of Chemo/Biosensing and Chemometrics and Peng Cheng Lab, China.

References

[1] C.C. Sanchez Zuleta, L.M. Giraldo Marín, J.F. Vélez Gómez, D. Sanguino Cotte, C.A. Vargas López, F.A. Jaimes Barragán, Evolution of the data mining and machine learning techniques used in health care: A scoping review, in: World Conference on Information Systems and Technologies, Springer, 2021, pp. 151–160.

[2] M. Azarm-Daigle, C. Kuziemsky, L. Peyton, A review of cross organizational healthcare data sharing, Procedia Comput. Sci. 63 (2015) 425–432.

[3] T. Wang, Y. Lei, Y. Fu, W.J. Curran, T. Liu, X. Yang, Medical imaging synthesis using deep learning and its clinical applications: a review, 2020, arXiv preprint arXiv:2004.10322.

[4] P. Sharma, R. Jindal, M.D. Borah, Blockchain technology for cloud storage: A systematic literature review, ACM Comput. Surv. 53 (4) (2020) 1–32.

[5] M. Dai, S. Zhang, H. Wang, S. Jin, A low storage room requirement framework for distributed ledger in blockchain, IEEE Access 6 (2018) 22970–22975.

[6] A. Palai, M. Vora, A. Shah, Empowering light nodes in blockchains with block summarization, in: 2018 9th IFIP International Conference on New Technologies, Mobility and Security, NTMS, IEEE, 2018, pp. 1–5.

[7] J. Benet, Ipfs-content addressed, versioned, p2p file system, 2014, arXiv preprint arXiv:1407.3561.

[8] S. Team, SWARM-storage and communication infrastructure for a self-Sovereign digital society, 2021.

[9] S. Wilkinson, T. Boshevski, J. Brandoff, V. Buterin, Storj a Peer-To-Peer Cloud Storage Network, Citeseer, 2014.

[10] R. Maani, S. Camorlinga, R. Eskicioglu, A remote real-time PACS-based platform for medical imaging telemedicine, in: Medical Imaging 2009: Advanced PACS-Based Imaging Informatics and Therapeutic Applications. Vol. 7264, SPIE, 2009, pp. 174–185.

[11] V. Rajkumar, M. Prakash, V. Vennila, Secure data sharing with confidentiality, integrity and access control in cloud environment, Comput. Syst. Sci. Eng. 40 (2) (2022) 779–793.

[12] S. Yao, R.V.J. Dayot, H.-J. Kim, I.-H. Ra, A novel revocable and identity-based conditional proxy re-encryption scheme with ciphertext evolution for secure cloud data sharing, IEEE Access 9 (2021) 42801–42816.

[13] P.J. Taylor, T. Dargahi, A. Dehghantanha, R.M. Parizi, K.-K.R. Choo, A systematic literature review of blockchain cyber security, Digit. Commun. Netw. 6 (2) (2020) 147–156.

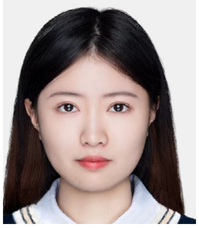
[14] Q. Zheng, Y. Li, P. Chen, X. Dong, An innovative IPFS-based storage model for blockchain, in: 2018 IEEE/WIC/ACM International Conference on Web Intelligence, WI, IEEE, 2018, pp. 704–708.

[15] D. Jia, J. Xin, Z. Wang, W. Guo, G. Wang, ElasticChain: Support very large blockchain by reducing data redundancy, in: Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data, Springer, 2018, pp. 440–454.

[16] H. Liu, X. Xiao, X. Zhang, K. Li, S. Peng, MIFS: A peer-to-peer medical images storage and sharing system based on consortium blockchain, in: International Symposium on Bioinformatics Research and Applications, Springer, 2021, pp. 336–347.

[17] J.S. Plank, S. Simmerman, C.D. Schuman, Jerasure: A Library in C/C++ Facilitating Erasure Coding for Storage Applications-Version 1.2. Vol. 23, Tech. Rep. CS-08-627, University of Tennessee, 2008.

Shaoliang Peng is a professor in the College of Computer Science and Electronic Engineering, Hunan University. His research interests include blockchain, biomedical big data, computer aided drug discovery, and high-performance computing.



Wenxuan Bao is a master student in the College of Computer Science and Electronic Engineering, Hunan University. She works on blockchain and big data.

Hao Liu is a master student in the College of Computer Science and Electronic Engineering, Hunan University. He works on blockchain and distributed system.

Xia Xiao is a master student in the College of Computer Science and Electronic Engineering, Hunan University. She works on blockchain.

Jiandong Shang has completed his Ph.D. from Xi'an Jiaotong University. He is the Director of the National Supercomputing Center in Zhengzhou, He has published more than 120 papers in reputed journals and has been serving as an editorial board member of repute.

Lin Han, Ph.D., associate professor of Zhengzhou National Supercomputing Center, graduate supervisor, member of CCF high performance computing Professional Committee of China computer society, distinguished expert of national information security engineering technology research center, high-level talents in Zhengzhou and 1125 innovative leaders in Zhengzhou. The main research interests are high-performance computing, domestic self-control, large-scale program optimization.

Shan Wang is an assistant professor in Faculty of Arts and Humanities and Institute of Collaborative Innovation, University of Macau. Her research interests include computational linguistics and linguistics.

Xiaolan Xie, female, professor and doctoral supervisor of the School of Information Science and Engineering, Guilin University of Technology. The current dean, research direction: cloud computing, big data.

Yang Xu, Ph.D., is an associate professor and deputy director of the Department of Cyberspace Security. He is a senior member of the Chinese Computer Society (CCF) and a member of the Blockchain Special Committee, and a member of the Blockchain Special Committee of the Chinese Society for Industrial and Applied Mathematics (CSIAM).