Available online at www.sciencedirect.com

ScienceDirect

Future Generation Computer Systems 00 (2025) 1–7

FGCS

A Decentralized Encrypted Data Sharing Scheme Based on Blockchain and IPFS

Mingxuan Chen[a], Guozi Sun[a,*]

[a]*School of Computer Science and Technology, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*

Abstract

Due to the explosive growth of data volume in the information age, users often store their data on remote servers managed by external service providers. Service providers are not always trustworthy, and user privacy becomes the primary concern. Although traditional encryption techniques can be used to protect outsourced data, they do not support effective retrieval of data, hindering data sharing. The emergence of searchable encryption technology provides an effective way to solve this problem, where users can directly perform complex operations such as searching and computing on encrypted data. However, most existing searchable encryption schemes delegate search operations to intermediaries such as cloud servers, in which case users may receive incomplete or even incorrect search results, and cloud server side may also experience single point of failure, privacy leakage, and other issues. The paper introduces blockchain technology to solve the semi trusted cloud server problem in traditional solutions, proposes a blockchain based encrypted data sharing scheme, and designs and implements a file sharing system based on Hyperledger Fabric and IPFS according to this scheme.Finally, we have demonstrated through sufficient experiments that our method is feasible and efficient.

Keywords: Data sharing, Cloud storage, Blockchain, Searchable Encryption, Hyperledger Fabric

1. Introduction

With the rapid development of cloud computing technology, more and more users choose to transfer data to the cloud to reduce the burden of local management. However, encrypted data can lead to reduced data availability, making it difficult to perform information search and other operations on ciphertext. Of course, users can choose to download all encrypted files and use the corresponding keys for decryption and search, but for large-scale data, this is not a suitable method [1].

Searchable encryption is an encryption primitive that can search uploaded encrypted data and is a promising technology. The encryption function provided by SE can prevent servers from stealing user personal data. The ciphertext search function allows users to search for specific keywords on encrypted data and return the target ciphertext file based on query requests, ensuring the privacy and security of user data without reducing query efficiency.

In most existing solutions, cloud servers are defined as ”honest but curious” entities. Honesty means that participants will not forge data, and servers will not engage in malicious operations such as attacks, cracking, or reverse engineering on the data uploaded by participants. Curiosity refers to a certain degree of curiosity on the server side towards the user’s raw data, and may bypass some security measures to directly access the user’s raw data. Malicious cloud servers may return incomplete or incorrect results, and users may also claim that the returned results are incomplete or incorrect after receiving the correct results. Therefore, the reliability of the scheme and the verifiability of the results cannot be fully guaranteed. In addition, each data owner has their own method for constructing indexes, but due to the lack of a unified verification mechanism that can be applied to all search schemes, information sharing between different organizations and individuals is limited.

The decentralization, integrity, invariance, and security of blockchain provide a solution to the problems existing in traditional searchable encryption. By adopting advanced information processing mechanisms such as distributed node consensus algorithms and encryption algorithms, blockchain technology can effectively record thereby effectively preventing

*Corresponding author
Email addresses: `2455602239@qq.com` (Mingxuan Chen), `sun@njupt.edu.cn` (Guozi Sun)

any form of illegal operations [2]. The searchable encryption technology based on blockchain can ensure the reliability and verifiability of the scheme, while promoting data sharing. In blockchain based methods, blockchain nodes use consensus mechanisms to automatically verify the elegitimacy and correctness of transactions, and any data will be permanently recorded.

Searchable encryption in static environments is not suitable for practical use, such as in medical and e-commerce scenarios where data needs to be constantly updated. Therefore, dynamic addition, update, or deletion functions are necessary. In short, forward security ensures that update algorithms should not disclose any useful information about keywords, while backward security ensures that subsequent search algorithms should not disclose index information of deleted files. Due to efficiency issues, most dynamic searchable encryption schemes do not support forward security. Our scheme introduces a blockchain based dynamic searchable encryption method that supports forward security [3]. This method improves its search efficiency in cloud environments while supporting forward security.

The existing SE can be divided into symmetric searchable encryption (SSE) and public key searchable encryption (PEKS). Searchable encrypted single user models are typically designed to encrypt and process single user data, thus having limitations in data sharing. Due to the fact that single user models mainly focus on individual data privacy protection and personalized processing, their encryption methods and computation processes typically revolve around a single user and cannot directly support data sharing and collaboration among multiple parties. In the current digital era, a large amount of sensitive data needs to be shared among different users, such as medical records, financial information, etc. Therefore, in the field of data sharing in multi-user scenarios, designing searchable encryption methods is crucial.The blockchain uses the Hyperledger Fabric consortium chain, while the cloud server uses the interstellar file system IPFS instead. Our scheme designs a BSSEShare data sharing system based on a decentralized cluster composed of consortium chain Fabric and IPFS, and tests and analyzes each operation step of the BSSEShare system.

The rest of this paper is organized as follows. The second section briefly introduces the relevant preparation work. The third section introduces the specific implementation of the plan. Then, the fourth part analyzed its security and performance. Finally, we summarized this paper in Section 5.

2. Related Work

2.1. Searchable Encryption

Traditional encryption techniques can only ensure the confidentiality of data and are difficult to perform effective calculations in a ciphertext state. Users who want to search for keywords can only download and decrypt a large number of files stored on the server one by one. The download and decryption process of all files not only consumes a lot of local resources, resulting in extremely low efficiency, data encryption protects data privacy but reduces user experience.

To address this issue, Song et al. [4] introduced the concept of searchable encryption in 2000. Searchable encryption allows cloud storage to search for encrypted data without knowing the relevant plaintext or keywords, enabling queries on encrypted data and ensuring the confidentiality and security of outsourced data. The basic idea is that the data owner encrypts the index and document collection before outsourcing them to the cloud server, which contains a set of keywords from all documents. To perform a search, the user generates a trapdoor and sends it to the cloud server, which returns documents related to the generated trapdoor.

The index can be static or dynamic, depending on whether the document collection can be changed. A static index is an index that never changes a collection of documents, meaning that no documents are added, deleted, or updated after the index is generated. These indexes are primarily used to demonstrate the effectiveness of the proposed searchable encryption scheme. If an index can effectively change with changes in the collection of documents, then the index is called a dynamic index. With the increasing dependence of users on third-party server data outsourcing, more data is being outsourced, leading to frequent changes in data collection. In practical applications, there may be a greater need for solutions that can support dynamic search of data.

Kamara et al. [5] first proposed a sublinear time DSSE scheme, which, although achieving fast deceleration and dynamic updates, has privacy leakage issues during the interaction process. Subsequently, Kamara et al. [6] proposed the first DSSE scheme that supports parallel queries. Although this scheme solves the problem of privacy information leakage, it increases the length of searchable ciphertext, leading to a decrease in search efficiency. Curtmola et al.'s scheme [4] authorizes multiple data users to query subsets and proposes the concept of multi-user SSE for the first time.

The traditional searchable encryption method is only applicable to situations where third parties are honest: the server honestly performs the search operation and only returns the true result after the data user pays the fee. However, in actual cloud environments, users only want to make payments after verifying the results returned by the server, and the server only wants to perform search operations after the user pays the service fee. In addition, it is possible that malicious servers may only return incomplete or even incorrect results after receiving fees. At the same time, users may also refuse payment and claim that the returned result is incomplete or incorrect, even if the returned result is correct [7]. Therefore, Li et al. [8] first introduced searchable encryption that supports blockchain, which can solve the problem of unfair service payments caused by this situation.

2.2. Blockchain

A blockchain based searchable encryption scheme helps detect malicious operations and supports fair payment mechanisms to punish malicious behavior. The decentralization, integrity, and invariance of blockchain can be well integrated with cloud computing. Through a fair payment mechanism, when servers and users execute according to the rules set by smart contracts, they can both receive fees and search results separately. When a cloud server is found to be dishonest, it will have its deposit confiscated instead of receiving fees. Like users, they cannot refuse to return results from cloud servers and refuse to pay fees, provided that the search results are genuine. Therefore, supporting blockchain searchable encryption helps to achieve fair payment services between cloud servers and users.

At present, blockchain based searchable encryption has been applied in multiple fields such as healthcare, e-commerce, outsourcing services, and vehicle social networks. Tahir et al. [9] first studied the SE scheme based on probability trapdoors for Hyperledger Fabric. In order to enhance the security of outsourced cloud data, the data is encrypted and stored on the blockchain. In order to search for encrypted data, this framework uses a privacy preserving searchable encryption scheme based on probability trapdoors, which hides search patterns, can resist distinguishability attacks, and ensure higher levels of security and privacy.

Hu et al. [10] designed a decentralized privacy protection search scheme, using smart contracts in Ethereum to introduce fairness, and designed a new smart contract so that every participant can be treated equally, and the interests of honest parties can be better protected. This scheme has many advantages, including support for dynamic updates, multiple users, fairness, and prevention of malicious server behavior.

Chen et al. [11] proposed a blockchain based vehicle social network (VSN) searchable public key encryption scheme. This scheme uses public key searchable encryption to solve the problem of identical symmetric keys that may occur in symmetric searchable encryption in VSN environments. It not only uses smart contracts to solve the problem of central server untrustworthiness, but also supports forward and backward security.

Chen et al.[12] proposed a blockchain based EHR shared searchable encryption scheme inspired by Hu et al.'s method. The previous plan placed both ciphertext and indexes on the blockchain, which may result in excessive computational burden on smart contracts and lower efficiency. This scheme adopts the system model shown in the above section, which only stores encrypted indexes on the blockchain, improving the efficiency of searching and verifying indexes.

Fu et al. [13] proposed a new blockchain based searchable multi cloud encryption scheme, which mainly optimizes the system model mentioned in the previous section. To address the issues of unstable access, easy leakage of user privacy, and difficulty in data recovery in a single cloud server model, a system model is first defined in multiple clouds, and multiple cloud service providers are combined to store data through a consortium chain. Then, encrypted documents and indexes are stored in the Interstellar File System (IPFS), and the hash value and IPFS address of the documents are stored in the blockchain, This reduces the server's control over the collection of encrypted documents.

2.3. IPFS

IPFS is a peer-to-peer hypermedia protocol that makes networks faster, safer, and more open. Simply put, it is a decentralized internet. Speaking of technology, it is a distributed file system based on distributed hash table DHT for content addressing, git model version management, Merkel object association, peer-to-peer technology, global namespace IPNS, and various technologies. The birth of blockchain was originally aimed at achieving decentralization, achieving consensus without a central organization, and jointly maintaining a ledger. It is not designed for high efficiency, low energy consumption, or scalability (if pursuing high efficiency, low energy consumption, and scalability, centralized programs may be a better choice).

The collaborative work of IPFS and blockchain can supplement the two major shortcomings of blockchain: low storage efficiency and extremely high cost; Cross chain collaboration requires coordination between different chains, making it difficult to coordinate.

Regarding the first issue, blockchain networks require all miners to maintain the same ledger, and each miner needs to keep a backup of the ledger locally. So, in order to ensure that the information stored in the blockchain is not tampered with, it is also necessary to keep a backup in the hands of each miner. This is very uneconomical. Imagine there are 10000 miners on the entire network now, even if we want to save 1M of information on the network, the storage resources consumed by the entire network are 10GB.

At present, there are also compromise solutions to alleviate this problem. When building decentralized application DAPP, the widely adopted approach is to only store hash values in the blockchain and store the information that needs to be stored in a centralized database. In this way, storage becomes a weak link in decentralized applications and a fragile link in the network.

IPFS proposes another solution: we can use IPFS to store file data and place the only permanently available IPFS address in a blockchain transaction, without having to place the data itself in the blockchain. Regarding the second issue, IPFS can assist various blockchain networks in transmitting information and files.

3. Scheme Description

Verifiable searchable encryption is a research field that extends SSE to ensure that customers always receive the correct files that match keywords and reject incorrect results. However, existing verifiable searchable encryption methods have the following security and functional limitations. Firstly, many methods [14] and [15] rely heavily on a particular validator for their validation process, assuming that the validator is a trustworthy or honest but curious entity, which can easily be affected by a single point of failure. Secondly, some existing verifiable SE schemes [16] are only designed for static data and are not suitable for practical application scenarios. One research direction of SSE is to consider deploying decentralized databases on the blockchain, which can avoid the problem of single points of failure. However, while index storage and search performed on smart contracts ensure fairness and achieve decentralization, the cost is high due to the need for smart contract platforms to pay for storing data and code, as well as executing code.

The method in this chapter is improved based on the verifiable searchable encryption method proposed by Zhang et al. [17]. Blockchain technology is introduced to synchronize the latest update status, ensuring the trustworthiness of the verification process. Only auxiliary data for the verification process is stored on the blockchain, greatly reducing synchronous communication and computation. In addition, the method in this chapter is based on the index construction of state chains and the search caching method to achieve optimized search computation and communication, further improving search performance. In addition, this chapter's scheme also achieves efficient verifiability of search results based on multi-set hash functions.

3.1. System Model

Figure 1 shows the designed system model, which consists of three types of entities: client (data owner), cloud server, and blockchain. The client represents the data owner, who generates encrypted databases (encrypted files and encrypted indexes) through encryption methods and outsources them to cloud servers. At the same time, verification information is generated and uploaded to the blockchain. Among them, the database can be dynamically updated by the data owner. In terms of blockchain, our scheme chooses to use the consortium chain Hyperledger Fabric, which is responsible for verifying whether the retrieval results returned by the cloud server are correct.

3.2. System Procedure

Our scheme introduces blockchain on the basis of traditional searchable encryption schemes to address the semi trusted problem of cloud servers. This method utilizes symmetric cryptographic primitives to improve search efficiency on large-scale data while satisfying forward security conditions. The auxiliary information required for verifying search results will be stored in the blockchain in the plan. When the client searches for the same keyword again in the future, the blockchain can return the previously cached results and transmit the updated index information back to the client, thereby improving efficiency and solving the trust problem in traditional solutions, fully utilizing the immutability of the blockchain.

Algorithm **??** depicts the initialization process pseudocode of a blockchain based dynamic searchable encryption method, which describes the collaboration process between the client, cloud server, and blockchain. This chapter proposes using state chains to achieve forward security under dynamic updates. Each keyword corresponds to a state chain, and all identifiers that match the keyword are stored in the positions derived from that state chain. When the client wants to search for keywords, they send the last status identifier st_{c+1} to the server. Starting from st_{c+1}, the server can obtain all previous states. Therefore, the server can traverse the state chain in reverse order and ultimately obtain all results. Due to the unidirectionality of the state chain, the server cannot obtain the next state from the current state, which ensures forward security properties. In order to further improve search and update efficiency, the states in the scheme are randomly generated instead of using permutation functions.

The verify process pseudocode of a blockchain based dynamic searchable encryption method is described in algorithm **??**. The scheme combines multiple sets of hash functions and blockchain to achieve efficient verifiability of retrieval results. The difference between multi set hash functions and standard hash functions is that they operate on multiple sets rather than strings. Specifically, a multi-set hash function takes a finite and unordered set of elements as input and maps them to a fixed length string, where one element can appear multiple times. In addition, the multi-set hash function is incremental, meaning that it can quickly update the hash value based on the previous hash value, making it easy and efficient to modify multiple sets.

3.3. Correctness Anaylsis

The correctness analysis of the scheme in this chapter mainly includes two aspects: the search algorithm should return all matching encrypted indexes; The validation algorithm should successfully validate the correct results while rejecting incorrect ones. For search algorithms, as long as the correctness of the trapdoor is ensured, the search algorithm can find all encrypted indexes corresponding to the keyword. State chain technology, as a storage structure on the server side, can treat each trapdoor as a state to search for all ciphertexts corresponding to previous and current states. When adding or deleting each node (keyword file identifier), a new node will be added to the corresponding keyword's state chain. Compared to the method proposed by Zhang et al., in order to avoid the situation where each search needs to be traversed from scratch, the server stores the current search result after each search, so that the final result can be integrated for the next search. In addition, the nodes in this chapter only record file identifiers, operation types, and the latest evidence, without retaining the status information of the previous node.

For the verification algorithm, this chapter adopts a combination of multi-set hash functions and blockchain technology to achieve the verification function. Each node contains a verification credential, which is generated by hashing the file identifier, latest status, and update times corresponding to the keywords selected by the client. When the client searches for keywords, the server returns the search results and corresponding verification credentials for the subsequent verification process.

4. An encrypted data sharing system based on Fabric and IPFS

This chapter proposes a file sharing system based on blockchain and searchable encryption to address the issues of poor security, privacy leakage, and low tamper resistance in data sharing. Considering the transparency and decentralization characteristics of blockchain, making data visible to everyone on the network is not suitable for enterprises that require permission management and control. Therefore, the system transfers user files to the interstellar file system IPFS for storage.

4.1. System architecture

The main function of the presentation layer in the system architecture is to receive user instructions or data inputs, submit them to the application layer for processing, and at the same time, be responsible for displaying the processing results to the user. The front-end development and compilation process of the project used VS Code tools. In terms of architecture, the system utilizes HTML, CSS, Vue, and Element UI to create a comprehensive display of the view layer. The Bootstrap framework is applied to static UI layout, while the Element component is responsible for presenting interface content, while utilizing Node.js and NPM for package management. Compared to other popular frameworks, Vue.js stands out for its bidirectional data binding feature and has effective network security protection strategies, such as measures to prevent SQL injection and XSS attacks, meeting the requirements of secure file management. Therefore, it has been selected as the basic framework for this project. To debug the project, the system uses Vue DevTools; To ensure routing security, Vue Router configuration is adopted; And achieve resource integration and deployment through Vue Loader and Webpack. In addition, the system relies on Vue CLI to build a standard project structure, utilizes Vuex to manage bidirectional data flow status, and uses Ajax asynchronous request technology Axios for data transmission to achieve file sharing. To address browser compatibility issues, the project adopts Babel technology; Eslint ensures standardized development syntax; Mock.js is used for preliminary data construction to improve development efficiency.

The logic layer in the system architecture focuses on operating on the data layer, abstracting raw data into logical data, and providing technical support for all functions of the application layer. This layer covers various interaction, authentication, storage, and encryption technologies. In this system, key technologies include Json Web Token (JWT), MySQL database technology, CouchDB state database, searchable encryption algorithm, and permission management control based on Casbin. For file storage, use the distributed content addressable file system IPFS. Any file stored in IPFS is uniquely identified by its content hash. Compared to centralized storage, using IPFS has many advantages, such as stronger fault tolerance, resistance to DoS attacks, and better scalability.

The persistence layer of the system architecture mainly covers Hyperledger Fabric and IPFS file storage systems. The IPFS file storage system is used to store files uploaded from user space and files shared by users; Hyperledger Fabric is used to store file identifiers, user behavior information, and indexes required for retrieving content. In terms of blockchain technology selection, the system has chosen Fabric because it adopts a modular design, allowing users to choose and configure different components according to their actual needs, thereby providing greater flexibility and scalability. In addition, Fabric supports multiple programming languages (such as Go, Node. js) to write smart contracts and provides powerful version control and update mechanisms. For the file storage part, a distributed content addressable file system IPFS is adopted. In IPFS, the unique identifier of any file is determined by the hash value of its content. Compared to centralized storage, the advantages of IPFS include stronger fault tolerance, resistance to DoS attacks, and better scalability.

4.2. User management

Figure provides a detailed description of the process of adding users to the system. The allocation of system permissions is based on a role-based permission management model, designed using the Casbin framework. When a user registers on their own, it is assigned as a regular user role by default, and subsequent role assignment tasks are handled by the system administrator. Once the user sets a password, the system will use the SHA-256 algorithm to encrypt the user's password and store the generated hash value in the blockchain. The SHA-256 algorithm, as a secure hash function, has advantages such as irreversibility, collision resistance, and high security. The hash value generated by it has a fixed and unique length, which can effectively protect the confidentiality of user passwords, provide a certain degree of data integrity and security, and ensure that user information is not easily stolen or tampered with during transmission and storage.

4.3. File management

Traditional SE solutions typically entrust an honest but curious CS to be responsible for storage and search management, which can lead to some endogenous issues such as single point of failure and untrustworthy results. That is to say, malicious CS may deviate from predefined specifications and engage in unauthorized tampering and deletion, which will make the search results unreliable. This chapter will seek to design a trustworthy multi keyword fuzzy retrieval method in a multi-user environment. To ensure data availability, IPFS is used to store encrypted data files in a decentralized, deduplicated, and tamper proof manner. The encrypted data files can be retrieved from multiple nodes simultaneously through content addressing to provide high throughput. Compared with existing IPFS based SE storage modes, in order to achieve a lightweight blockchain architecture, the system model in this chapter uploads encrypted data files from IPFS instead of most modes to the blockchain. IPFS adopts a decentralized architecture, where data is dispersed and stored on multiple nodes, avoiding a single point of failure. Its content addressing technology is based on hash value to locate data, rather than traditional location-based addressing methods, effectively ensuring the integrity and security of data. In addition, IPFS has fine scalability, which can easily expand storage capacity and improve performance, maintaining the stability of the system in long-term operation.

When department managers publish public documents, the system obtains the current logged in user information and verifies whether their role is a department administrator. If the user attribute is not a department manager, the system administrator needs to approve and grant file sharing permissions. After permission verification, the department administrator selects the users or departments who want to share files and authorizes them to share their own keys. Subsequently, the system encrypts the files using keys and salt values, and uploads the encrypted files to IPFS for storage. At the same time, the system uses trigram method and Bloom filter to generate indexes, and uploads encrypted indexes to the blockchain for permanent storage.

4.4. Permission management

The permission control part of this system adopts the RBAC model of the Casbin framework, which is used to achieve fine-grained permission control for users. Casbin is a powerful access control framework, whose RBAC model manages the relationships between users, roles, and permissions by clearly defining policies and models, effectively achieving secure access control of resources. By utilizing Casbin's RBAC model, access control rules can be easily managed and executed to enhance the security and data protection level of applications. The permission control process after the user initiates an operation request is shown in figure **??**.

Firstly, during the system initialization phase, the system reads the user-defined model, which includes the definitions of four parts: Access Request, Access Policy, Matcher, and Effect. Meanwhile, Casbin utilizes the Adapter module to read access policy instances, abbreviated as access policies, from the data source. Casbin decouples the storage and reading of access policies into independent adapter modules. By using different adapters (such as File, MySQL, etc.), access policies can be read from different data sources. When a request arrives, it will be split according to the definition in the model, and the matcher will match with the access policy based on the model rules. The matcher supports strong matching and also supports fuzzy matching through Stub Function and RoleManager. Stub Function allows users to customize matching rules and pass in custom functions to the matcher for complex matching. In practical applications, a request may match multiple access policies. It is necessary to aggregate multiple access policy results and ultimately return the result of whether the request is allowed.

4.5. Searchable Encryption Analysis

This section first compares the designed multi keyword fuzzy searchable scheme with other related schemes [10], [11], [18], [19], mainly comparing whether it supports the following attributes: blockchain, multi-user environment, multi keyword retrieval, fuzzy retrieval, and IPFS storage, as shown in table **??**. In terms of trustworthy search, except for MKSSMDO which relies on centralized CS, other solutions have introduced blockchain to ensure credibility. Both MKSSMDO and the proposed scheme in this chapter consider multi keyword search with multiple data owners, and only the proposed scheme in this chapter supports fuzzy search. Some schemes essentially consider single user models, which suffer from complex key distribution and high communication overhead in multi-user environments. For storage issues, some solutions store encrypted data files in a centralized model, which inevitably reduces the availability of data. Hu et al. directly uploaded encrypted files and encrypted indexes to the blockchain, and this system model would suffer from high communication and storage costs. Although the paper mentioned the use of IPFS storage, no explanation was given. Only this chapter demonstrates how to enhance data availability by using IPFS storage.

This section first compares the time cost of this chapter's scheme and MKSSMDO in the index construction algorithm through experiments. Figure **??** shows the time and cost required for two schemes to generate indexes and trapdoors as the number of keywords in the data file changes. To ensure fairness in the experiment, a keyword dictionary was predefined to store a set of keywords for all files, and the dictionary size was fixed at 200. In addition, the number of hash functions is set to 3, 6, and 9 respectively, and the size of the Bloom filter is set to 192 bits. As shown in Figure **??**, when the number of keywords in the data file varies between 10 and 80, the time cost of index construction in this chapter's scheme shows a linear growth trend. This is because each keyword is decomposed into multiple triples and added to the index. Therefore, as the number of keywords increases, the number of generated triples also increases linearly, thereby increasing the complexity and size of the index. In contrast, when the fixed keyword set size is 200 and the hash function is 3, the time required for this chapter's scheme is significantly less.

Next, this section compared the time cost of the method proposed in this chapter with MKSSMDO in the trapdoor generation algorithm through experiments. Figure shows the time and cost required for two schemes to generate trapdoors as the number of keywords in the data file changes. It should be noted that the implementation of MKSSMDO only considers the time cost of exponentiation in bilinear mapping, while ignoring the time cost of each keyword hash operation. In our scheme, the time cost of all operations in the trapdoor generation process is fully considered. Due to the fact that the efficiency of the trapdoor generation algorithm only depends on the number of keywords and the size of the Bloom filter, the time cost of the method in this chapter will increase linearly with the increase of the number of search keywords (within the range of 10 to 80). The time cost of generating trapdoors in this chapter's scheme is almost the same as that of MKSSMDO. In addition, the time cost of index generation and trapdoor generation in this chapter's scheme slowly increases with the increase of the number of hash functions.

Overall, the proposed scheme can support fuzzy retrieval and trusted retrieval, but its index generation and search performance are slightly lower than the MKSSMDO scheme, mainly due to the use of trigrams to build indexes. The trigram index has good fuzzy matching ability, which can quickly find similar strings and even recognize them when there are certain differences. This makes trigram indexing very effective for handling spelling errors, case mismatches, or partial matches. In scenarios such as full-text search and pattern matching, trigram indexing can significantly improve query performance and result accuracy. However, trigram indexes also have some drawbacks. Firstly, it consumes a significant amount of storage space as it requires storing all trigram combinations for each string, which may cause issues when dealing with large-scale datasets. Secondly, due to the large number of trigram tokens, the cost of building and maintaining trigram indexes is relatively high, and every time data is updated or inserted, trigram tokens need to be recalculated.

4.6. Blockchain throughout and latency

The functional testing and subsequent performance testing experiments in this section were conducted in an experimental environment consisting of 6 machines, with 4 machines serving as servers for IPFS and Fabric, and the other 2 machines acting as file uploaders and file downloaders respectively. The server's machine ran on the Ubuntu 20.04 system, while the client's machine ran on the Windows 10 system. The server is equipped with an Intel Xeon E-2225G processor, the client machine is equipped with an AMD Ryzen 7 5800H with Radeon Graphics processor, and all machines are equipped with SSD solid-state drives.

The performance testing of the system is mainly divided into three main aspects: interstellar file system IPFS, blockchain Fabric, and searchable encryption methods. For the testing of searchable encryption methods, including index generation performance, trapdoor generation performance, and keyword retrieval. In order to evaluate the performance of IPFS and Fabric in a cluster environment, this paer deploys the server of the system on four hosts within the same LAN. Each host is deployed with Fabric and IPFS, with Fabric configured with 5 order nodes (2 on one host and 1 on the other) and 4 peer nodes. Each peer node is divided into 2 organizations, and IPFS also deploys 4 nodes to form a network together. In addition, to compare the performance differences with IPFS during file upload and download processes, MinIO was also deployed on these four hosts.

5. Conclusion

Traditional searchable encryption schemes typically assume that the remote server is an honest but curious entity, meaning it never attempts to deviate from the specified protocol. However, in practical scenarios, malicious servers may return incomplete or even incorrect results. Blockchain is a decentralized ledger that has the characteristics of openness, transparency, and tamper resistance, ensuring the public and trustworthy retrieval and verification processes. Therefore, our scheme combines blockchain technology with searchable encryption technology to design an encrypted data sharing scheme based on consortium chain and IPFS.Our scheme utilizes blockchain technology and multi-set hash functions to achieve verifiability of search results, and utilizes state chain technology and search result caching methods to achieve forward security. The trigram algorithm and Bloom filter were used to achieve multi keyword fuzzy retrieval, and the IPFS interstellar file system was used to ensure the robustness of encrypted data files.

References

[1] S. Liu, L. Chen, G. Wu, H. Wang, H. Yu, Blockchain-backed searchable proxy signcryption for cloud personal health records, IEEE Transactions on Services Computing 16 (5) (2023) 3210–3223.

[2] H. Liu, Y. Ming, C. Wang, Y. Zhao, S. Zhang, R. Lu, Blockchain-assisted verifiable certificate-based searchable encryption against untrusted cloud server for industrial internet of things, Future Generation Computer Systems 153 (2024) 97–112.

[3] H. Yu, S. Liu, L. Chen, Y. Gao, Blockchain-enabled one-to-many searchable encryption supporting designated server and multi-keywords for cloud-iomt, Journal of Systems Architecture 149 (2024) 103103.

[4] D. X. Song, D. Wagner, A. Perrig, Practical techniques for searches on encrypted data, in: Proceeding 2000 IEEE symposium on security and privacy. S&P 2000, IEEE, 2000, pp. 44–55.

[5] S. Kamara, C. Papamanthou, T. Roeder, Dynamic searchable symmetric encryption, in: Proceedings of the 2012 ACM conference on Computer and communications security, 2012, pp. 965–976.

[6] S. Kamara, C. Papamanthou, Parallel and dynamic searchable symmetric encryption, in: Financial Cryptography and Data Security: 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers 17, Springer, 2013, pp. 258–274.

[7] H.-B. How, S.-H. Heng, Blockchain-enabled searchable encryption in clouds: A review, Journal of Information Security and Applications 67 (2022) 103183.

[8] F. Li, K. Liu, L. Zhang, S. Huang, Q. Wu, Ehrchain: a blockchain-based ehr system using attribute-based and homomorphic cryptosystem, IEEE Transactions on Services Computing 15 (5) (2021) 2755–2765.

[9] S. Tahir, M. Rajarajan, Privacy-preserving searchable encryption framework for permissioned blockchain networks, in: 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), IEEE, 2018, pp. 1628–1633.

[10] S. Hu, C. Cai, Q. Wang, C. Wang, X. Luo, K. Ren, Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 792–800.

[11] L. Chen, W.-K. Lee, C.-C. Chang, K.-K. R. Choo, N. Zhang, Blockchain based searchable encryption for electronic health record sharing, Future generation computer systems 95 (2019) 420–429.

[12] X. Chen, S. Xu, T. Qin, Y. Cui, S. Gao, W. Kong, Aq–abs: Anti-quantum attribute-based signature for emrs sharing with blockchain, in: 2022 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2022, pp. 1176–1181.

[13] S. Fu, C. Zhang, W. Ao, Searchable encryption scheme for multiple cloud storage using double-layer blockchain, Concurrency and Computation: Practice and Experience 34 (16) (2022) e5860.

[14] J. Zhu, Q. Li, C. Wang, X. Yuan, Q. Wang, K. Ren, Enabling generic, verifiable, and secure data search in cloud services, IEEE Transactions on Parallel and Distributed Systems 29 (8) (2018) 1721–1735.

[15] Z. Shi, X. Fu, X. Li, K. Zhu, Esvsse: Enabling efficient, secure, verifiable searchable symmetric encryption, IEEE Transactions on Knowledge and Data Engineering 34 (7) (2020) 3241–3254.

[16] A. Soleimanian, S. Khazaei, Publicly verifiable searchable symmetric encryption based on efficient cryptographic components, Designs, Codes and Cryptography 87 (1) (2019) 123–147.

[17] Z. Zhang, J. Wang, Y. Wang, Y. Su, X. Chen, Towards efficient verifiable forward secure searchable symmetric encryption, in: Computer Security–ESORICS 2019: 24th European Symposium on Research in Computer Security, Luxembourg, September 23–27, 2019, Proceedings, Part II 24, Springer, 2019, pp. 304–321.

[18] H. Li, H. Tian, F. Zhang, J. He, Blockchain-based searchable symmetric encryption scheme, Computers & Electrical Engineering 73 (2019) 32–45.

[19] H. Yin, Z. Qin, J. Zhang, L. Ou, F. Li, K. Li, Secure conjunctive multi-keyword ranked search over encrypted cloud data for multiple data owners, Future Generation Computer Systems 100 (2019) 689–700.