World Wide Web (2023) 26:2749–2766
https://doi.org/10.1007/s11280-023-01148-0

On-chain repairing for multi-party data migration

Wenchao Jiang[1] · Weiqi Dai[2] · Qiwen Lv[2] · Dongjun Ning[3] · Sui Lin[1]

Received: 15 July 2022 / Revised: 5 November 2022 / Accepted: 1 February 2023 /
Published online: 9 May 2023

Abstract
Blockchain can be used to solve the problem of mutual trust between different institutions. However, when migrating data from a traditional system to a blockchain system, the order of data transactions is difficult to determine and the data transactions on the chain can not be modified. Therefore, it is necessary to build an on-chain repairing mechanism for multi-party data migration. In this paper, an on-chain repairing system for multi-party data migration is designed to realize the multi-party data migration, the block data repairing, and the data auditing. The multi-party data migration is utilized to determine the order of data transactions, and the order determining during data transaction is solved by setting up transaction pools and sorting transactions of the same institution or different institutions. The controlled data repairing strategy is utilized to repair the error data in the blockchain based on the chameleon-hash algorithm. The data repairing audit strategy is used to ensure the controllable data repairing. Compared with the Hyperledge Fabric, the additional cost for multi-party data migration of our method is not more than 10%.

Keywords Blockchain · Online migration · Transaction sequencing · Data repairing · Block auditing

1 Introduction

With the rapid development of information technology, many user friendly applications are emerging. While enjoying the convenience brought by these applications, users are inevitably faced with the problems of data communication between different platforms. At present, the applications using traditional centralized technologies has been

This article belongs to the Topical Collection: *Special Issue on Privacy and Security in Machine Learning*
Guest Editors: Jin Li, Francesco Palmieri and Changyu Dong

✉ Weiqi Dai
wqdai@hust.edu.cn

[1] School of Computer Science and Technology, Guangdong University of Technology, 510006 Guangzhou, China

[2] School of Cyber Science and Engineering, Huazhong University of Science and Technology, 430074 Wuhan, China

[3] Taotall Technology Development Co., Ltd, 510635 Guangzhou, China

confirmed to exist many security problems. Exchanging data between different organizations might cost a long time. For example, each bank has its own clearing system and data need to be exchanged between banks when inter-bank transactions happen. Because the clearing system between different banks has different data agencies and involves more sensitive data, it usually takes a lot of time to clear the data.

Furthermore, the organizations do not trust each other, and so it is difficult to achieve data sharing among multiple organizations safely and reliably. At present, data sharing is mainly achieved through the multi-agency centralized data trading platforms. But centralized data trading platforms may also do bad things. It is difficult to achieve safe and reliable data sharing among multiple institutions. Users need to switch information between applications in different institutions, which increases learning costs and the risk of private data leakage. Users need to use different applications in different usage scenarios but data between applications of different institutions is not inter connected. So users often need to switch applications with different requirements. Due to the different usage habits produced by different manufacturers, users need to resubmit information for registration and relearn the application operation method. On the one hand, users resubmit their identity data to a new third party centralized platform can increase the risk of their private data leakage. On the other hand, users need to learn the operation methods of the new application which results in the poor user experience.The traditional application platform has the problem of leaking user privacy. Due to lacking the corresponding technical means to supervise the user behaviour, users' privacy data misuse incidents continue to appear. The platform sells the private information of users on the platform to other companies for profits. There is no effective supervision technology to regulate corporate behaviors facing deliberately selling user privacy data. Such data leakage events are difficult to avoid, there are no effective technical ways to prevent the leakage of privacy data from continuing propagation. On the other hand, the database of the centralized platform is easy to be attacked by hackers. User information may be resold by the database managers due to the poor management of data rights.

Blockchain technology [1] is developing rapidly and it can be an efficient way to realize the safely interaction between institutions. Due to the decentralized nature of blockchain technology [2], multiple institutional nodes can achieve mutual trust and complete security data sharing with fast data interaction at the same time [3]. By migrating applications to the blockchain platform, it is possible to solve the trust problems existing in the centralized business systems between different institutions. The traditional centralized business system is inefficient and requires a lot of manpower to ensure its reliability, and the market needs to transfer centralized business to a more efficient blockchain platform. The decentralized nature of the blockchain system can solve the trust problem between different institutions. The data needs to reach an agreement before it can be uploaded, so it is difficult for hackers to modify it [4]. Dynamic migration of multi-party data can migrate applications from the traditional centralized environment to the decentralized environment of the blockchain. It can ensure that the service of the application is not interrupted and the entire migration process has a low impact on system operation.

However, there are still the following difficulties and challenges in achieving multi-party data migration on the blockchain system.

- Current migration ways cannot be directly applied to the blockchain environment. The current data migration solution is mainly aimed at the traditional centralized scenario. The data needs to reach consensus among various nodes. When data is migrated from multi-party centralized nodes, the order of the data transaction itself is difficult to be determined.

- Repairing the block data is difficult and the cost of data repairing is high. There may be errors during the data migration process. The traditional centralized organization can modify the data in the database directly, but this data management method has hidden security risks. The user's data is always stored in a centralized organization, and it cannot prevent the organization from doing evil. It is easy to be obtained or modified by hackers. The current blockchain system does not support data security repairing functions. At the same time, when deploying a hard fork to roll back the data, there will be a huge price to pay. For example, the decentralized autonomous organization (DAO) incident uses a hard fork to reduce the losses caused by hackers. Eventually, it not only consumes a lot of time, but also affects the entire Ethereum community, which making Ethereum [5] split into two chains.

There is currently no effective data migration solution in blockchain scenarios. With the increasing demands of users for more secure and convenient applications, the demand for migrating applications to the blockchain platform is also increasing. At present, the application of traditional centralized technology has the above mentioned problems. At the same time, the traditional data migration method cannot be directly used for blockchain data migration. Therefore, it is currently necessary to design an on-chain repair system for multi-party data migration. In this paper, an on-chain repairing system for multi-party data migration is designed to realize the multi-party data migration, the block data repairing and the data auditing. The multi-party data migration is proposed to determine the order of data transactions. The order determining during data transaction is solved by setting up transaction pools and sorting transactions of the same institution or different institutions. The controlled data repairing strategy is utilized to repair the error data in the blockchain based on the chameleon-hash algorithm. The data repairing audit strategy is used to ensure the controllable data repairing.

The rest of this paper is organized as follows. The related works is given in In Section 2. The problem definition and system design and are presented in Section 3. Section 4 analyzes the security performance of the designed system. Prototype implementation and the relevant evaluation is reported in Section 5, and finally, we conclude our work in Section 6.

2 Related works

Blockchain technology is widely used in the digital currency [6], commerce [7], medical treatment [8], digital transaction [9], finance and other aspects. The advantage using blockchain technology is that the blockchain platform can achieve user mutual trust without the introduction of a third party. With the increasing demand of users for data reliability, blockchain traceability technology [10] is widely used in the supply chain to ensure that users can query the logistics information of commodities and ensure the authenticity of commodity sources. Some data that should not be tampered can also be stored using the blockchain platform, such as the e-invoice, subway, financial insurance, and other industries [11]. While blockchain technology provides convenience for users, it also brings security problems that have not appeared in traditional centralized applications [12]. At present, the data security repairing mainly focuses on the rollback of transaction data, so as to realize the security repairing for erroneous data. When the data is wrong, the blockchain community rolls back the transaction to eliminate the huge losses caused by the hacker. Due to the different operating environments of transaction data, data security repairing methods can be divided into the use of hard fork rollback data in the blockchain environment and rollback data in the centralized environment. Data rollback is currently mainly achieved through a hard fork to realize the repairing data on the chain. A hard fork means that after the block code changes, the block eneration rules are no longer forward compatible and the data received by the nodes after the code upgrade is different from the node that has not been upgraded. Therefore, block data can be repaired in a hard fork for the security repairing of the data on the chain. The consensus of the blockchain code can be manually set to encourage the miner nodes in the blockchain network to reach consensus above the data on the chain. Most of the nodes on the chain will refuse the wrong block data.

Once an error is found in an exchange in a centralized environment, the error data can be repaired by rolling back the abnormal data in the exchange's database. A hacker attack occurred on Binance, which is the largest cryptocurrency exchange. After freezing the abnormal transactions, Binance chose to roll back all the abnormal transactions to reduce the losses caused by hackers. However, this method can only be executed within the Binance Exchange. The data rolled back is the data in the internal database of the exchange and the data transactions in the trading platform have not sent to the blockchain. At the same time, due to the impact of the rollback operation, people have reduced their trust in the platform. The rollback behavior itself violates the blockchain spirit, which will also affect the market value of the trading platform. At the same time users who conduct trading operations may also be injured by mistake. It can be concluded that there is currently no perfect and mature solution for data repairing. In response to the increasing demand for data repairing by users, a more convenient solution is needed.

In the traditional centralized environment, data migration is also required. When an application migrates data from an old platform to a new platform, it is necessary to use a data migration method to transfer the application data and to achieve a smooth transition between the old and new platforms without affecting the normal application services of the system. There are two major methods to migrate data which include middleware and data migration tools. Middleware method develops a set of data structures for data transfer, obtains data from the data source, and converts them into the data structures for data transfer. Then the middleware processes the data of the specified data structure and imports it into the database of the target application. Using middleware for data migration has some advantages. It can be more flexible for data migration and the efficiency of data migration is faster. Compared with using data migration tools, using middleware for data migration does not need to rely on third-party tools and data does not need to send into the third-party tool software, so it is more secure and reliable. There are many kinds of data migration tools, such as In-fomover, Flyway, Liquibase, Phinx, and Oracle Warehouse. Alibaba cloud open source data migration tool DataX is an offline synchronization tool for heterogeneous data sources. It can synchronize data between heterogeneous data sources such as MySQL, Oracle, and other databases. Using data migration tool for data migration is easy. It is simple to operate but the data migration process will be limited by the data migration tool.

The feature that blockchain is not editable also brings inconvenience to people in certain scenarios. Users should have the right to edit and delete their data. At the same time, for some sensitive data, such as bank transaction amount and other data, once there is an error, they need to be able to repair these data. The chameleon-hash algorithm [13] and signature mechanism were first proposed by Krawczyk and Rabin [14] to realize a hash function to find collisions for a given hash with an undeniable data signature mechanism. The chameleon-hash algorithm has a public key and a private key. The person with the public key can calculate the hash value and the person with the private key can quickly find the hash collision. For those who do not have a private key, it is difficult to find a hash collision so the hash function is collision resistant. The researching [15] is used in the field of blockchain to realize data repairing. For example, Ateniese designed an editable blockchain based on the chameleon-hash [16]. In their design, the standard hash functions are replaced with chameleon-hashes. By finding a hash collision, a new block with a constant hash value was constructed to enable editable data function for the entire block. A fine-grained block data editing method based on the chameleon-hash algorithm uses attribute based encryption [17] for finegrained access control of encrypted data. Deuber et al. proposed the first efficient redactable blockchain for the permissionless setting [16] that is easily integrable into Bitcoin [18]. Xiangji Cai proposed an editable blockchain that can protect users' privacy [19]. Li et al. Proposed an efficient and secure outsourcing of differentially private data publishing with multiple evaluators [20, 21]. This method do not need to rely on complex encryption tools or any trusted institutions by using the consensus mechanism based on spatial proof. At the same time to ensure the traceability of the data on the chain, they use the ring signature scheme [22–24].

3 System design

3.1 System overview

We designed an on-chain repairing system for multi-party data migration. The main roles in the system include users, contract issuers that initiate data repair requests, and trusted nodes. The entire process of the on-chain repair system for multi-party data migration is shown in Figure 1. According to the analysis of the entire process in the on-chain repairing system for multi-party data migration, the following attacks may occur.

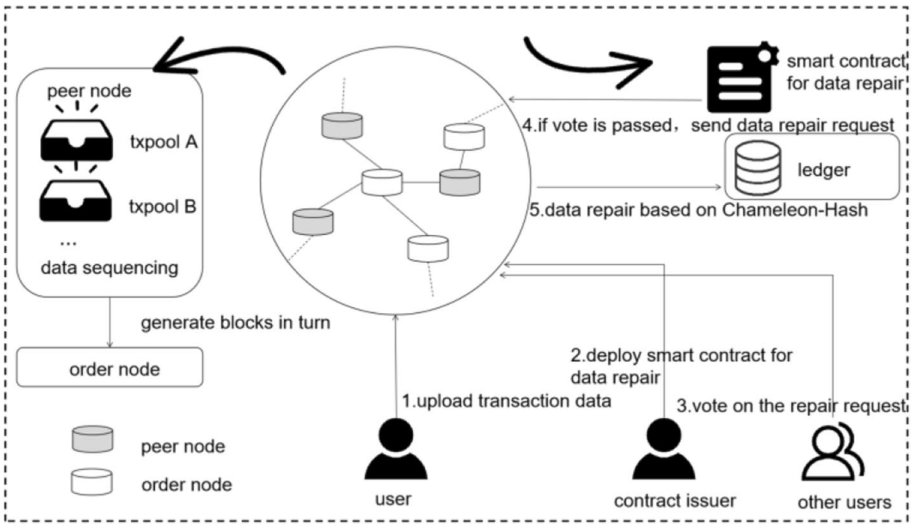
peer node
txpool A
txpool B
...
data sequencing
generate blocks in turn
order node
peer node
order node
smart contract for data repair
4.if vote is passed，send data repair request
ledger
5.data repair based on Chameleon-Hash
2.deploy smart contract for data repair
1.upload transaction data
3.vote on the repair request
user
contract issuer
other users

Figure 1 System overview

- **Data Acquisition Stage.** At this stage, users mainly send transaction data that needs to be uploaded to the blockchain system. The user obtains the data to be uploaded from the local database, deals with the data, and sends it to the system. The main security threat at this stage is that users may upload false transaction data for profits.

- **Data Sorting Stage.** At this stage, the blockchain system processes transaction data sent by users. Different transaction pools are set up for each institution and transactions in this institution are accepted as transaction recipients. Cross-bank transactions and non-cross-bank transactions are sorted in the transaction pool. The main security threat at this stage is that users may upload out-of-order transactions, which affects the order of transactions in the transaction pool.
- **Repair Preparation Stage.** This stage is mainly the stage where the contract issuer prepares for the data repair request on the chain. The contract issuer deploys smart contracts, uploads block repair data, and initiates a vote on whether to perform the repair operation. Users in the blockchain network can vote on the repair request. The main security threats faced at this stage include that the contract issuer may deliberately release the wrong data repair plan to achieve the purpose of tampering with the data on the chain. Users may be bribed and maliciously vote on the repair request to manipulate the repair of the data on the chain.
- **Data Repair Stage.** The nodes in blockchain verifies the data repair request sent by the user and checks the voting results. If the verification is correct, the data repair operation is performed. The main security threats faced at this stage include that users maliciously send a large number of data repair requests and launch denial of service attacks to consume resources in the system.

3.2 Assumptions

- The security assumptions of an on-chain repair system for multi-party data migration are as follows.
- The nodes in the alliance chain will not be bribed or do evil. They get the submitted block and repair the block which is determined to be repaired.
- The on-chain repair system for multi-party data migration can work normally with functions such as block generation, consensus, and block data editing. It can deal with the transactions sent to the blockchain network.
- Users or nodes will not cause data loss due to network reasons in network transmission.

3.3 Threat model

- In summary, the security threats facing the system may come from the following aspects.
- Threat 1: Users may upload a large amount of transaction data to the chain in a short time, consuming a large amount of communication resources in the blockchain network. They may launch a denial of service attack to consume the resources in the system, causing other normal users to fail to upload data in time.
- Threat 2: Users may upload false transaction data for profits.
- Threat 3: The contract issuer may deliberately release the wrong data repair plan to achieve the purpose of tampering with the data on the chain.

- Threat 4: Users may be bribed and maliciously vote on repair requests to manipulating the repair of data on the chain.

3.4 System designing

The system mainly includes multi-party data migration strategy, block data controlled repairing strategy, and block data repairing audit strategy. Figure 2 shows more details about the designing and the subsequent parts will introduce the specifics of each module in detail.

3.4.1 Data migration

Platform users transfer data from multiple centralized databases to a decentralized blockchain platform. This process requires a data migration module to ensure the safety and reliability of the data migration process. The data migration module includes a transaction pool buffer submodule, a transaction sequencing submodule for the same institutions, and a transaction sequencing submodule for different institutions. Figure 3 shows more details about it.

The transaction pool buffer module is used to cache and preprocess the acquired transactions. All transactions will be sent to the transaction pool of the corresponding institution. The system sets up additional counters to record the number of transactions currently packed into the block. When transactions enter the transaction pool, each transaction needs to be classified. For each different institution, different trading pools are set up. After each transaction is acquired, it is necessary to judge the type of transaction and place the transaction into the corresponding transaction pool. When packaging

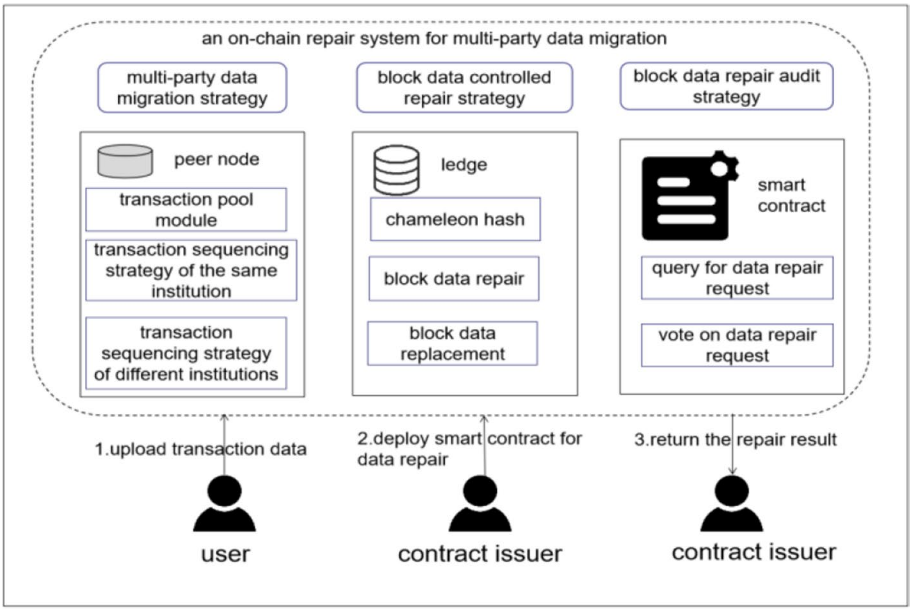
an on-chain repair system for multi-party data migration
multi-party data migration strategy
block data controlled repair strategy
block data repair audit strategy
peer node
transaction pool module
transaction sequencing strategy of the same institution
transaction sequencing strategy of different institutions
ledge
chameleon hash
block data repair
block data replacement
smart contract
query for data repair request
vote on data repair request
1.upload transaction data
2.deploy smart contract for data repair
3.return the repair result
user
contract issuer
contract issuer

Figure 2 System design

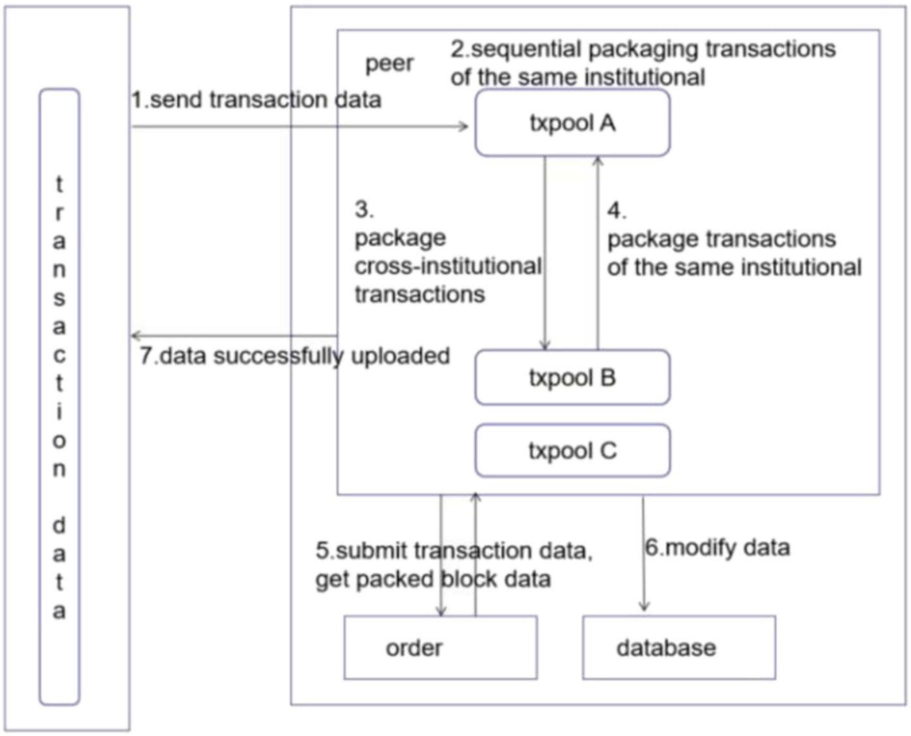
transaction data
peer
2.sequential packaging transactions of the same institutional
1.send transaction data
txpool A
3. package cross-institutional transactions
4. package transactions of the same institutional
7.data successfully uploaded
txpool B
txpool C
5.submit transaction data, get packed block data
6.modify data
order
database

Figure 3 Design of data migration

transactions, it is necessary to switch between transaction pools. When processing a transaction, the system needs to check whether the transaction belongs to an existing transaction pool. If there is no transaction pool corresponding to the current transaction, the system will create a new transaction pool. The transaction pool needs to be initialized first. Depending on the type of the first transaction entering the transaction pool, different treatments are required. If the transaction happens in one institution, transactions happened before this transaction should be dealt with first. Otherwise, the system will put transactions that belong to the bank with the serial number before the currently processed transaction to the transaction pool. When new data is sent to the blockchain system, the transaction pool needs to add new transactions. When transactions are added to the transaction pool, judgments must be made between transactions to prevent duplicate transactions from being added. The system will check the attribute fields of transaction A and transaction B. If the transaction input party, transaction output party, transaction amount, transaction input party serial number, and transaction output party serial number are all the same, it is considered to be the same transaction. Otherwise, it is considered a different transaction.

The transactions of the same institution are sorted according to the order in which the transactions occur. According to the transaction serial number of the transaction itself, the transactions are sorted in the transaction pool corresponding to the transaction receiving institution. When adding transactions of the same institution to the transaction pool, the system will check whether transactions have been added to the corresponding institution's transaction pool. After the current transaction pool obtains the right to package, the transactions in the transaction pool can be packaged.

The order of transactions in different institutions needs to be determined by the system. During the packaging process, it is necessary to determine the type of each transaction. When encountering a cross-institution transaction, it will automatically be sent to the transaction pool of the institution that initiated the transaction. The system will record the information of the transaction and check the serial number N of the transaction. Then all transactions that occurred before N need to be processed and packaged in sequence. If cross-institutional transactions are encountered during the packaging, it will switch to the transaction pool of the current transaction for packaging. The transaction pool class is constructed to handle crossinstitution transactions. The current cross-bank transactions in the transaction pool are recorded in Txbf. When the current transaction is a cross-bank transaction, the Inbf function needs to be executed fifirst to fifind the position in the cross-bank transaction list and the current transaction will be deleted from the list of inter-bank transactions.

3.4.2 Block data repairing

The main process of the data repair module is as follows and Figure 4 shows more details about it.

The chameleon-hash algorithm is used to generate block hash values to ensure that the block hash remains unchanged when the block data is modified, and further ensure the continuity of block data. The block data editing module is mainly implemented in the transaction pool. After the node finds a special transaction in the transaction pool that the requests for block data security repair, the node will first verify the special transaction. The verification information includes whether the vote in the smart contract passed. The system gets the data by the communication between the blockchain platform and the smart contract. The blockchain system interacts with smart contracts to obtain the information about whether the vote is passed. According to the modification information submitted by the user, the system reconstructs new block data. First the system locates the block data that

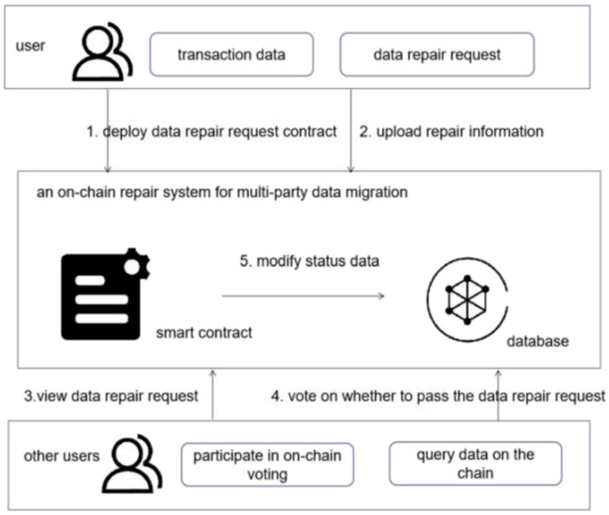
user
transaction data
data repair request
1. deploy data repair request contract
2. upload repair information
an on-chain repair system for multi-party data migration
5. modify status data
smart contract
database
3.view data repair request
4. vote on whether to pass the data repair request
other users
participate in on-chain voting
query data on the chain

Figure 4 Design of block data repairing

needs to be modified by the block number. Then the system locates the transaction data that needs to be modified by the hash of the transaction. After the block data is modified and repackaged, the hash of the block is unchanged because of the trapdoor.

3.4.3 Data repair auditing

The blockchain system needs to reconstruct the block data according to the repair information submitted by the user. The blockchain system needs to obtain the repair information of the block in the smart contract, which is submitted by the user. Figure 5 shows more details about it.

After the node obtains the user's data, it needs to process the data and construct a new block according to the modified information. The node obtains the current block data from the world state and gets the block data that needs to be modified. According to the data repair request submitted by the user to the smart contract, the node sends the repair information after the repair is completed to the smart contract. After the vote is passed, the smart contract needs to send a special transaction requesting data repair to the blockchain network. After receiving the special transaction of the data recovery request, the blockchain network processes the special transaction of the data recovery request. The data structure of the special transaction includes the block data information to be modified. After receiving special transactions, the nodes in the blockchain network perform the data repair operations according to the special transactions to achieve world state synchronization. According to the special transaction requesting data repair, the nodes determine the type of request and execute the corresponding data repair module. The user needs to deploy the smart contract applying for data repair and uploads the block data repair scheme to the smart contract. The nodes of the entire network vote on whether to pass the smart contract about the user data repair request. At the same time the block node calculates the new hash code through the trapdoor and uploads the modified data to the smart contract. After the users successfully vote for the smart contract, the smart contract needs to send a data repair request to the blockchain network. The data repair request includes the address of the smart contract deployed and the type of operation the user applies for modification. Data repair requests are sent to the blockchain network through special transactions. Among them, the special transaction is a new type of transaction introduced in the blockchain. By setting up new data structures for parsing and adding processing logic to the data within the

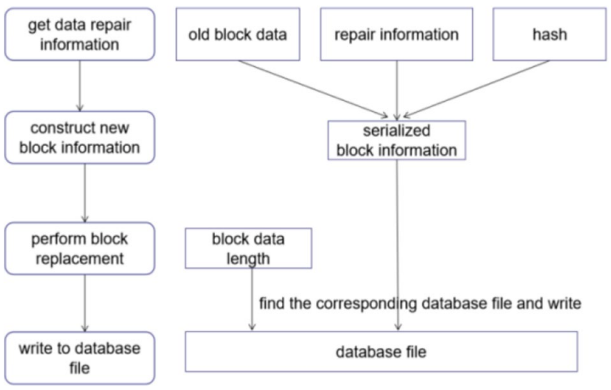
get data repair information
construct new block information
perform block replacement
write to database file
old block data
repair information
hash
serialized block information
block data length
find the corresponding database file and write
database file

Figure 5 Design of data repair auditing

blockchain system, user's data repair requests can be dealt with by sending transaction messages to the blockchain network.

4 Security analysis

In the process of data migration from a traditional centralized system to a blockchain platform, the order of transactions on the chain may be inconsistent with the actual order due to network fluctuations and differences in the local environment of different institutions. Because of the difficulty in reaching an agreement on the chain, a data migration module is constructed in the on-chain repair system for multi-party data migration. By constructing transaction pools of various institutions, each transaction pool is sorted according to the transaction serial number and each transaction pool generates blocks in turn. When dealing with the cross-institution transactions, they are transferred to the corresponding institution's transaction pool. The previous transactions based on the serial number should be packaged. After processing previous transactions, the system turns to the original institutional transaction pool and package transactions to generate new blocks.

In the on-chain repair system for multi-party data migration, users can repair the erroneous data that has been on the chain, so there is a possibility of malicious abuse of the data repair function. In response to the problem of the malicious abuse of data repair function, a data repair audit module is designed. Smart contract are deployed to record data repair requests. On the one hand, all data repair requests initiated by users can be queried and users in the system can play the role of mutual supervisor. On the other hand, once the data repair request initiated by the user is found to be abnormal, the data repair request deployed by the user will be denied, which ensures that the malicious data repair request initiated by the user cannot be executed and makes data repair on the chain controllable.

The denial of service (DoS) attacks may occur in the on-chain repair system for multi-party data migration. In this system, malicious users may send a large number of transactions to the blockchain network in a short time, resulting in normal users unable to upload transaction data. Malicious users send a large number of data recovery requests in a short time and the communication resources in the blockchain network are consumed maliciously, resulting in normal users unable to complete data recovery operations.

First, for the first type of attacker who uploads a large amount of transaction data in a short period and prevents the transaction data sent by normal users from being uploaded to the chain in time, a blocking mechanism is set up to solve this problem. For each user, there is an organization that the user belongs to and a transaction pool is set for each organization. Transaction pools generate blocks in turn. Therefore, even if a malicious user uploads a large amount of data in a short time, the data of other users can be guaranteed to be packaged in time. At the same time, malicious users will be restricted from accessing the data in the alliance chain, thereby reducing the phenomenon of malicious users in the blockchain network. Second, the second type of attacker sends a large number of data recovery requests in a short time, causing the communication resources in the blockchain network to be maliciously consumed. It results in the phenomenon that normal users cannot complete the data recovery operation. In response to this problem, by limiting the number of data recovery requests sent by each smart contract, it is ensured that an attacker cannot send a large number of data recovery requests in a short time. At the same time, when users need to apply for the repair of data on the chain, they first need to deploy a smart contract. After the vote is passed, the smart contract sends a data repair request to the blockchain network, so other users who pass the vote can send the data repair request normally.

The case of uploading false data includes users may upload false transaction data for profits or the contract publisher deliberately publishes the wrong data repair scheme to achieve the purpose of tampering with the data on the chain. The system designs a data transaction confirmation mechanism for users who may upload false transaction data for profits. After the transaction organization confirms the transaction, they will upload the transaction to the blockchain network and the transaction will be sent to the corresponding transaction pool. Therefore, if the user uploads nonexistent data, the fake transaction cannot be uploaded. At the same time, if the user uploads transaction data that is maliciously tampered with, there are mechanisms to deal with it. On the one hand, due to the transaction confirmation mechanism, the transaction that the transaction receiver does not confirm can not be uploaded to the chain. On the other hand, for the wrong transaction on the chain, the user can initiate a data repair request to modify the wrong transaction on the chain. Aiming at the problem of deliberately uploading wrong data repair requests, a contract voting verification module is set up. In order to tamper with the data on the chain, the contract issuer deliberately releases the wrong data repair plan. At the same time, the users on the chain vote on whether to repair the contract. If the vote is not passed, the data repair plan will fail. Therefore, if the user uploads the wrong data repair request, it will be discovered by other users on the chain during the deployment of the smart contract.

Malicious voting means that the user may be bribed to maliciously vote on the repair request to achieve the purpose of manipulating the data repair on the chain. The user may intentionally vote to affect the normal data repair process of other users. By setting the voting threshold, users need to pay a certain deposit before they can vote. At the same time, a credit evaluation mechanism is established. Once the user conducts malicious voting, the user's credit points are deducted. Once the points are low, the user can not vote.

5 Prototype implementation and evaluation

Based on the various needs of users and the difficulties of data migration, we designed an on-chain repair system for multi-party data migration. The specific design goals are as follows.

5.1 System module analysis

First, the system needs to be able to edit block data. Once users find data errors during the migration process, the data can not be repaired in the traditional blockchain platform. Because the traditional blockchain is a decentralized system and the block edit function is not supported, it cannot meet the user's needs. The design of this system takes it into account that the data of the blockchain need to be repaired. So the block data can be edited based on the chameleon-hash algorithm to meet user's needs in our platform and Figure 6 shows more details about it.

Second, the system needs to sort the migrated data. The on-chain repair system for multi-party data migration needs to realize the function of the data migration module. When data is migrated from multiple centralized data sources to a decentralized blockchain platform, the data can be out of order. Because multiple centralized data sources migrate data at the same time and the network status can be bad, the order of data can

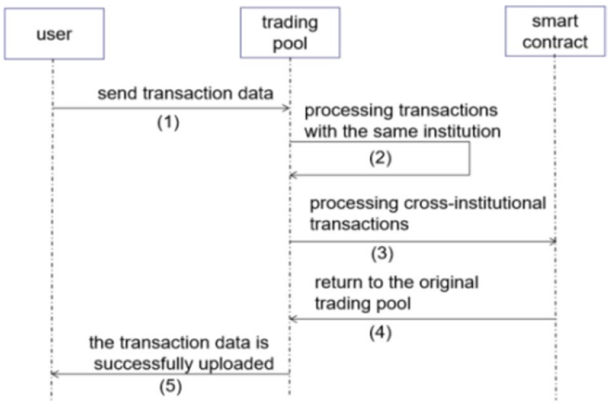
user
trading pool
smart contract
send transaction data
(1)
processing transactions with the same institution
(2)
processing cross-institutional transactions
(3)
return to the original trading pool
(4)
the transaction data is successfully uploaded
(5)

Figure 6 Graph of data repair process

be wrong if the time of data arrival is the key to determine the data sequence. Therefore, this system has designed a data migration strategy to obtain data from multi-party data sources and temporarily store them in the transaction pool. Different institutions have their transaction pools and multiple transaction pools generate blocks alternately to ensure the order of data during data migration. Figure 7 shows more details about it.

Third, the system needs to implement the function of the block data repair audit. Users in the blockchain network can check the data repair requests on the chain to prevent users from using the data repair function to do evil. This system uses smart contracts to record data repair requests on the chain. The contract issuer initiates a data repair request by deploying a smart contract and sends the edited repair data to the smart contract. At the same time, the smart contract initiates a vote. If the vote is passed, the data repair request is sent. In the repair operation, the block data is repaired successfully. If the vote fails, the block data repair fails. Through the above mentioned block data repair audit mechanism, it can ensure that the data on the chain can be repaired and the data repair can be audited. Figure 8 shows more details about it.

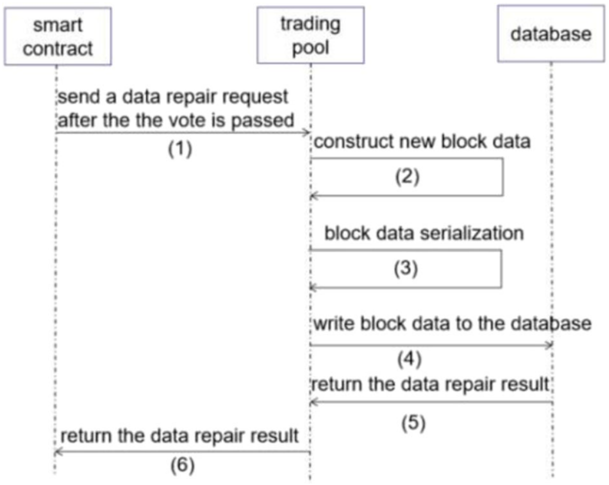
smart contract
trading pool
database
send a data repair request after the the vote is passed
(1)
construct new block data
(2)
block data serialization
(3)
write block data to the database
(4)
return the data repair result
(5)
return the data repair result
(6)

Figure 7 Graph of data migration process

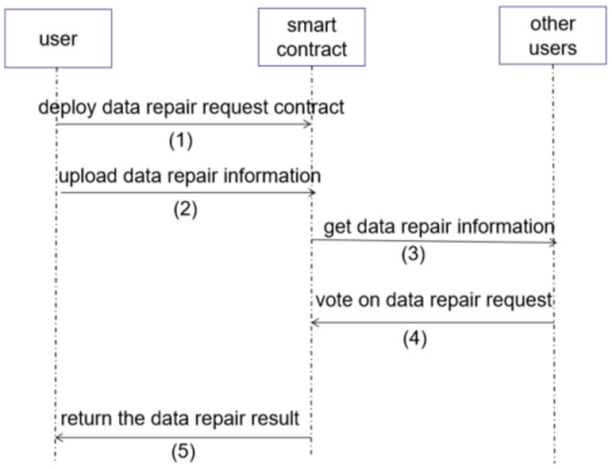
user
smart contract
other users
deploy data repair request contract
(1)
upload data repair information
(2)
get data repair information
(3)
vote on data repair request
(4)
return the data repair result
(5)

Figure 8 Graph of data repair audit process

5.2 Performance testing

In order to test the performance of the system, this paper carries out a set of tests. Performance testing is divided into performance testing of data repair module and data migration module.

A. Data Migration Module. For the block data migration module, this paper repeatedly executes multiple tests and use different numbers of data sets to test the time impact of the additional code on the system, as shown in the following Figure 9.

It can be seen that as the number of data sets increases, the time of the data migration module also increases and the time loss of the data migration function module shows a linear increase. During the sorting process, all transactions in the transaction pool

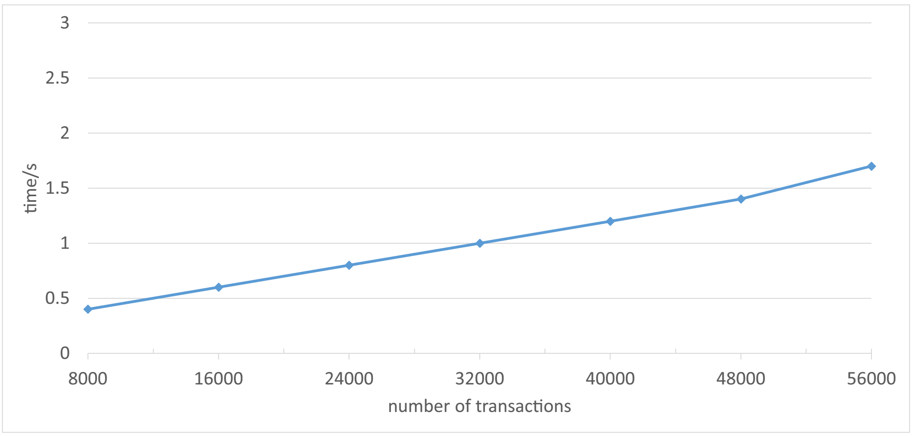

Figure 9 Data migration module performance test

will be parsed and sorted. The transaction pool itself needs to be maintained, so it will cause time loss. The test results show that the time loss caused by the newly added module to the system is within the acceptable range. Analyzing the time consumption of the data migration module, the main time consumption is concentrated on the cost of maintaining the transaction pool. The system designs a function to handle new transactions. When there are unprocessed transactions in the transaction pool, it is determined whether there is a transaction pool of the corresponding institution in the system's transaction pool list. When there are no pending transactions in the transaction pool, unlock other transaction pools to generate blocks.

B. Data Repair Module. For the data repair module, different numbers of data sets are used for performance testing and the time consumed is recorded at the same time. The specific time consumption is shown in the Figure 10.

It can be seen that as the number of datasets continues to increase, the time spent on the data repair module also increases. The time loss of the data repair function module shows a linear increase. The data repair module mainly takes time to write the repaired block data into the database. By obtaining the information and create the modified block, the modified block data is uploaded to the chain to achieve the purpose of repairing the data on the chain.

Since the number of transactions per second (TPS) of Hyperledge Fabric is around 250 and the time loss caused by the new functions of the system are concentrated on the data repair module and the data migration module, the actual time loss per second is less than 0.009 s. So the additional cost brought by the on-chain repair system for multi-party data migration is within 10%. The tests are mainly carried out according to the function module. For the data security repair function and data migration function, the performance tests of multiple sets of data are carried out separately and the final test results were compared with the Hyperledge Fabric. Compared with the Hyperledge Fabric, the additional cost brought by the data migration function and the data repair function for multi-party data migration is within 10%.

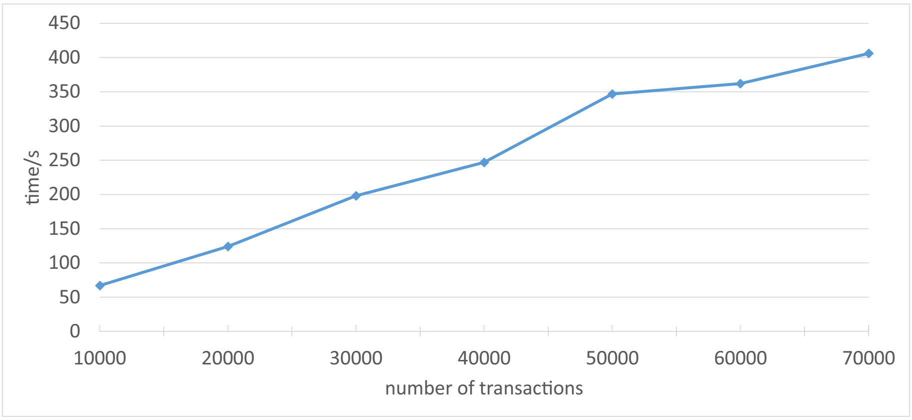

Figure 10 Data repair module performance test

6 Conclusion

This paper designs an on-chain repairing mechanism for multi-party data migration. An on-chain repairing system for multi-party data migration is implemented to ensure that users can migrate data to a blockchain platform, repair wrong data on the chain, and audit data repair requests. The system designed a data migration strategy to obtain data from multi-party data sources and temporarily store them in the transaction pool. Different institutions have their transaction pools and multiple transaction pools generate blocks alternately to ensure the order of data during data migration. The design of this system takes into account that the data of the blockchain may need to be repaired, so the block data can be edited based on the chameleon-hash algorithm to meet the needs of users. Users in the blockchain network jointly supervise the data repairing requests on the chain to prevent users from using the data repairing function to do evil. This system uses smart contracts to record data repairing requests on the chain. After analyzing the security of the system and completing the tests of the system performance, the results show that the system can achieve the expected goal. Compared with the Hyperledge Fabric, the additional cost for multi-party data migration of our method is not more than 10%.

The system realizes the migration of multi-party data on the chain, ensuring that the block can be repaired and the repair requests are audited on the chain. At the same time there are still some shortcomings. The current implementation scheme is designed based on the alliance chain assuming in which the nodes in the entire blockchain network are credible and they will not do evil in this blockchain network. However, it is inevitable to meet the public chain scene. The current design scheme has not yet applied this situation and it needs to be improved in the future.

Author contributions Wenchao Jiang designed the system and the framework of this paper, Qiwen Lv conducted the implementation, experiments and tests, Wenchao Jiang, Weiqi Dai, and Qiwen Lv wrote the paper together, Dongjun Ning provided the application and test scenario, and Sui Lin and Weiqi Dai check and proofread the paper.

Funding This study was funded by the Guangdong Science and Technology Plan Project (2019B010139001), Guangzhou Science and Technology Plan Project (201902020016), Guangdong Natural Science Fund Project (2021A1515011243), Yunfu Science and Technology Plan Project (S2021010104) and the National Key R&D Program of China under Grant No. 2019YFB2101700, the National Natural Science Foundation of China No. 62072202.

Data availability No

Declarations

Human and animal ethics No

Ethical Approval and Consent to participate Not applicable

Consent for publication We grant the publishing rights to the journal of World Wide Web.

Competing interests All the authors declare there are no any competing interests

References

1. Sidhu, J.: Syscoin: A peer to peer electronic cash system with blockchain based services for e-business. In Proceedings of the 26th International Conference on Computer Communication and Networks. IEEE, pp. 1–6. (2017)
2. Zheng, Z., Xie, S., Dai, H., Chen, X., Wang, H.: An overview of blockchain technology: Architecture, consensus, and future trends. In Proceedings of 2017 IEEE International Congress on Big Data. IEEE Computer Society, pp. 557–564. (2017)
3. Casino, F., Dasaklis, T.K., Patsakis, C.: A systematic literature review of blockchain based applications: current status, classification and open issues. Telematics Inf. **36**, 55–81 (2019)
4. Nguyen, G., Kim, K.: A survey about consensus algorithms used in blockchain. J. Inform. Process. Syst. **14**(1), 101–128 (2018)
5. Shurov, A., Malevanniy, D., Iakushkin, O., Korkhov, V.: Blockchain network threats: The case of pow and ethereum. In Proceedings of the 19th International Conference on Computational Science and Its Applications. Springer, pp. 606–617. (2019)
6. Ronen, E., Shamir, A., Weingarten, A., O'Flynn, C.: Iot goes nuclear: creating a zigbee chain reaction. IEEE Secur. Priv. **16**(1), 54–62 (2018)
7. Yao, Q..: A systematic framework to understand central bank digital currency. Sci. Chin. Inform. Sci. **61**(3), 033101:1-033101:8 (2018)
8. Azaria, A., Ekblaw, A., Vieira, T., Lippman, A.: Medrec: Using blockchain for medical data access and permission management. In Proceedings of 2nd International Conference on Open and Big Data. IEEE Computer Society, pp. 25–30. (2016)
9. Hasan, H.R., Salah, K.: Proof of delivery of digital assets using blockchain and smart contracts. IEEE Access. **6**(448), 439–465 (2018)
10. Mitani, T., Otsuka, A.: Traceability in permissioned blockchain. IEEE Access. **8**, 573–21588 (2020)
11. Atzei, N., Bartoletti, M., Cimoli, T.: A survey of attacks on ethereum smart contracts (sok). In Proceedings of the 6th International Conference on Principles of Security and Trust International Conference. Springer, pp. 164– 186. (2017)
12. Conti, M., E, S.K., Lal, C., Ruj, S.: A survey on security and privacy issues of bitcoin. IEEE Commun. Surv. Tutorials. **20**(4), 3416–3452 (2018)
13. Ateniese, G., de Medeiros, B.: On the key exposure problem in chameleon hashes. In Proceedings of the 4th International Conference on Security in Communication Networks. Springer, pp. 165– 179. (2004)
14. Krawczyk, H., Rabin, T.: Chameleon signatures. In Proceedings of the Network and Distributed System Security Symposium. The Internet Society, pp. 143– 154. (2000)
15. Camenisch, J., Derler, D., Krenn, S., Phls, H.C., amelin, K., Slamanig, D.: Chameleon hashes with ephemeral trapdoors and applications to invisible sanitizable signatures. In Proceedings of the 20th International Conference on Practice and Theory in Public Key Cryptography. Springer, pp. 152– 182. (2017)
16. Ateniese, G., Magri, B., Venturi, D., Andrade, E.R.: Redactable blockchain or rewriting history in bitcoin and friends. In Proceedings of 2017 IEEE European Symposium on Security and Privacy. IEEE, pp. 111– 126. (2017)
17. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute based encryption for fine grained access control of encrypted data. In Proceedings of the 13th ACM Conference on Computer and Communications Security. ACM, pp. 89–98. (2006)
18. Deuber, D., Magri, B., Thyagarajan, S.A.K.: Redactable blockchain in the permissionless setting. In Proceedings of the 2019 IEEE Symposium on Security and Privacy. IEEE, pp. 124– 138. (2019)
19. Cai, X., Ren, Y., Zhang, X.: Privacy protected deletable blockchain. IEEE Access. **8**, 6060–6070 (2020)
20. Li, J., Ye, H., Li, T., Wang, W., Lou, W., Thomas Hou, Y., Liu, J.: Rongxing Lu: efficient and secure outsourcing of differentially private Data Publishing with multiple evaluators. IEEE Trans. Dependable Secur. Comput. **19**(1), 67–76 (2022)
21. Li, J., Huang, Y., Wei, Y., Lv, S., Liu, Z.: Changyu Dong, Wenjing Lou: Searchable symmetric encryption with Forward Search privacy. IEEE Trans. Dependable Secur. Comput. **18**(1), 460–474 (2021)
22. Bresson, E., Stern, J., Szydlo, M.: Threshold ring signatures and applications to ad-hoc groups. In Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology. Springer, pp. 465–480.
23. Wu, Z., Li, G., Shen, S., Cui, Z., Lian, X., Chen, E., X. Su.: Constructing dummy query sequences to protect location privacy and query privacy in location-based services. World Wide Web (ISSN 1386-145X). 24 (1): 24–45 (2021)

24. Wu, Z., Shen, S., Zhou, H., Li, H., Lu, C.: A Basic Framework for Privacy Protection in Personalized Information Retrieval. J. Org. User Comput. **33**(6), 1–26 (2002)