Received October 22, 2019, accepted November 14, 2019, date of publication November 27, 2019, date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2956157

EduRSS: A Blockchain-Based Educational Records Secure Storage and Sharing Scheme

HONGZHI LI AND DEZHI HAN, (Member, IEEE)
College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China

Corresponding author: Dezhi Han (dzhan@shmtu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672338 and Grant 61873160.

ABSTRACT Accurate and complete educational records are a valuable asset for people. In recent years, educational records have been digitized. However, there are still two key challenges that have not been resolved. One is to achieve secure and privacy-preserving storage of educational records, while another concern how to realize the sharing of educational records and ensure the security of the sharing process. In this paper, we propose EduRSS, a blockchain-based storage and sharing scheme for educational records is proposed, which combines blockchain, storage servers, and cryptography techniques to create a reliable and safe environment. In our proposal, the blockchain technology is used to ensure the security and reliability of data storage, while the smart contracts on the blockchain are used to regulate the process of storage and sharing. More precisely, the off-chain storage servers store the original educational records in encrypted form, while the hash information of the records is stored on the blockchain. The off-chain records are anchored periodically with the hash information on the blockchain to ensure the security of data storage. Cryptography techniques are utilized to handle records encryption and messages digital signature. To assess the effectiveness of EduRSS, we designed and tested a proof of concept of this scheme. The relative security analysis shows that EduRSS is safe and has a lower computational cost than that of the CP-ABE and the MA-CPABE schemes.

INDEX TERMS Educational records, blockchain, smart contract, secure storage and sharing.

I. INTRODUCTION

Nowadays, for individuals, educational records describe their education-specific processes. These records are of great importance for the further study and career of an individual. The essential attribute of records is primitive, which enables records to restore the actual historical situation, so these records are essential to the students themselves, education institutions and potential employers. With the development of information technology, educational records have been digitized. Compared with the traditional paper physical records, digital records are stored on the storage medium which has a high degree of variability, hence such records could be easily modified during the processes of storage, transmission, and sharing.

Centralized storage and management mode is usually adopted, which makes systems that use this mode vulnerable to various attacks. Besides, the records of different educational stages are stored in separate storage servers of education institutions and these storage servers are usually designed to allow access only by internal staff, without any form of interoperability. Moreover, a server failure could easily cause a data loss or leakage. Therefore, in order to protect personal information, institutions usually adopt security policies to restrict the access and sharing of records. However, there is a lack of secure and effective record sharing mechanisms among institutions. For instance, students may experience difficulties when they transfer from one institution to another, while still preserving their completeness of courses from the previous institution. In cases when students apply for postgraduate programs in another education institution, this process requires the provision of past educational records, such as, courses grades, award certificates and so on, by which time they have completed the previous stage of education, left the previous education institution, etc., but usually, such students have no access to the online educational records system. In this case, they have to visit the previous institution to request a paper copy of their educational records,

The associate editor coordinating the review of this manuscript and approving it for publication was Yi Qian.

but this process takes a lot of energy and time. Moreover, if the storage server confronts data tampering, the educational records are rewritten, then people cannot know his/her true educational experience from these records. This management mode lacks flexibility and efficiency, it does not allow to realize neither the security and reliability of records nor the sharing and utilization of records. For example, in February 2011, a document containing sensitive information on 13,000 students of Chapman University was accidentally publicly disclosed [1]. Again, in September 2018, Jiangsu News Radio reported an important case of student identity information leaked or used by illegal enterprises, for tax evasion at Wade College of Changzhou University. With an initial estimate of more than 2600 students affected [2]. A decentralized solution based on Blockchain [3], [4], which enables to distribute educational records between different education institutions, could be effective in solving the above problems. We remark that public blockchain is not suited in this case, because educational records are related to personal privacy and contain sensitive information, such as family address, age, contact details, etc. Moreover, even if the institutions put encrypted data on the public blockchain, it still will expose their operation situations and statistical data. Therefore, the consortium blockchain maintained by educational institutions is a suitable choice.

Motivated by the above challenges and issues, a secure storage and sharing scheme for educational records is proposed, named EduRSS. As far as we know, this is one of the few works that apply blockchain to practice in the field of education. The main contributions of this study can be summarized as follows:

- A novel scheme is proposed, which integrates educational records storage and sharing among education institutions enabled by blockchain, storage servers and smart contracts. The blockchain is responsible for ensuring the security and auditability of the data, the smart contract is used to define the permissions of the records and to regulate the behaviors of the member nodes.
- A consortium blockchain is constructed, which is formed by institutions to record educational data in a verifiable and immutable ledger to guarantee data auditability. The encrypted data is stored in the storage server and their hash is put on the blockchain.
- An experimental system based on the EduRSS is implemented, the security analysis shows that the EduRSS is safe, and experiment results demonstrate that the EduRSS has a lower cost than that of the CP-ABE and MA-CPABE schemes.

The rest of this paper is organized as follows. In Section II, we give an overview of related work. In Section III, we introduce the background necessary to understand the proposed scheme, such as blockchain transactions and smart contracts. In Section IV, we describe the design of our proposal, while in Section V, we provide the details of the proposed scheme based on blockchain. In Section VI, we describe the implementation of EduRSS working prototype and analyze the security of the proposed scheme. Finally, In Section VII, we present the conclusions and future research directions for this work.

II. RELATED WORKS

The blockchain is applied in several domains and acts as a trusted data storage technology. This technology is often used for information secure storage and information traceability, because of its decentralized and anti-tampering characteristics. For this reason, a significant research effort is currently dedicated to this technology. Furthermore, this technology is finding more and more fields of application.

In the field of digital copyright, several schemes have been put forward. The authors of [5] present a digital watermark management system based on smart contracts, this system adopts non-repudiation of smart contract and blockchain, to track the use records of files, and recover losses copyright holders suffered. A blockchain-based scheme for digital rights management called DRMChain is proposed [6], in which two isolated blockchain applications are employed to respectively store plain and encrypted summary information of original digital content. Considering the big size of content such as image or video, a flexible external storage scheme of plain/encrypted content is proposed. More precisely, the original content or files are stored in the database whilst the information hash is stored in the chain. In the healthcare field, the work of [7], [8] presents medical records secure storage and service framework to solve the problems of personal medical data privacy protection and secure sharing. In the work of [9], blockchain and IoT-based health management scheme is proposed to serve diabetes patients, this system can monitor patients remotely and be able to sense their health, thus, warn them about potentially dangerous situations. The authors of [10], [11] put forward a secure data storage solution in the power energy field, they all utilize the blockchain technology to strengthen the protection of energy data against cyber-attacks. In transportation and supply chain management, the blockchain technology can also be applied [12]. In fact, since traditional electric vehicle charging systems are centralized by design, these systems are vulnerable to Distributed Denial of Service (DDoS) attacks to the central charging server, then a blockchain-based charging system has been proposed to solve these problems, this system provides the security of key, secure mutual authentication and also provides efficient charging. In [13], in order to ensure safe and efficient traffic operation, while assessing the trustworthiness of crowdsourced data, the authors use smart contracts on the blockchain, to eliminate single authority over such systems, and finally put forward a secure and real-time traffic event detection solution. Besides, there is some research work about improving the security and reliability of communication data in the network of vehicles [14], [15]. In the emerging field of Internet of Things (IoT), the blockchain technology has been widely used in various scenarios, the authors of [16] design a security architecture that applies the blockchain technology and the software-defined network (SDN) to improve the ability of IoT to deal with sophisticated cyber-attacks. In [17]–[19], the research work mainly focuses on the security of data storage on IoT, and [20] proposes an electric business model by using the blockchain technology for the IoT.

Blockchain technology can also be applied in the education field. Several research works have been carried out, and some educational institutions have begun to apply such technology to practical management. The work of [21] proposes a blockchain-based high education credit platform, with the concept of the European Credit Transfer and Accumulation, this platform should include a globally trusted, decentralized higher education credit that can offer a globally unified viewpoint for students and higher education institutions. Therefore, blockchain technology becomes an inevitable choice. The authors of [22] propose applying blockchain for secure storage of personal educational reputation and reward. The authors of [23] introduce the features and advantages of blockchain technology and explore some of the current blockchain applications for education. In addition, the study presents several potential issues of applying blockchain technology in education, such as eliminating the possibility of modifying educational records for legitimate reasons. The University of Nicosia is the first institution that uses the blockchain technology to manage students' certificates received from MOOC platforms [24]. Sony Global Education company has developed technology using blockchain for open sharing of academic proficiency and progress records [25]. An online learning platform based on blockchain has been developed by the Massachusetts Institute of Technology (MIT) [26]. Argentinian College has applied the blockchain technology to educational data management [27]. The researchers of Holberton School in San Francisco announced that they can help employers verify academic credentials by using the blockchain technology [28].

However, in the field of education, most of the above-mentioned projects are still in the conceptual stage. In fact, we have not found any literature about the specific implementation of blockchain technology in the field of education. In this work, we describe the design and implementation of the proposed EduRSS in detail.

III. PRELIMINARIES

In order to better understand the proposed scheme, in this section, we provide the necessary background concerning the main technologies on which this scheme is based. Table 1 presents some of the notations used in this paper.

TABLE 1. The descriptions of notations.

Notation	Description
$EDI_{(X)}$	Education institution X
$PK_{(X)}$	Public key of the institution X
$SK_{(X)}$	Private key of the institution X
$CT_{(P)}$	Ciphertext of the object P
$PT_{(P)}$	Plain text of the object P
$S_{(P)}$	Hash value of the object P
CA	Certificate Authority (CA) node
$CON_{(CA)}$	Connection object to the CA node
$U_{(X)}$	Attributes set of the institution X
k	Security parameters
$OF_{(X)}$	Official information of the institution X
$AuInfo_{(X)}$	Encrypted and packaged institutional information, received by the CA node
$VRT_{(X)}$	Vote results for institution X joining process
$CERT_{(X)}$	Certificate of institution X to join the EduRSS
$RD_{(X)}$	Educational record of the institution X
$TxId$	ID of a blockchain transaction
TX	Transaction object from the blockchain
$BlockAddress_{(X)}$	The address of institution X on blockchain

A. TRANSACTIONS ON BLOCKCHAIN

The transaction is the actual business data stored on the blockchain, the block is formally composed of several confirmed transactions, these transactions will be organized together in the form of Merkle Tree [5]–[8]. Essentially, the transaction is a signed packet that allows some digital coins to be transferred from one account to another. Besides, it can be used to store crucial information, and trigger the execution of smart contracts. In detail, in this work we use Ethereum where the structure of a transaction is mainly composed of the following parameters: an account that initiated the transaction, the received address, gas, gas price, blockHash, block number and the additional data field, etc. The detail descriptions of the transaction parameters are shown in Table 2.

TABLE 2. The descriptions of the blockchain transaction parameters.

Parameters	Descriptions
blockHash	the hash of the block in which the transaction is located
blockNumber	the block height in blockchain
from	the account that initiated the transaction
to	the address or account of the transaction recipient
hash	the hash value of the transaction
input	the hash value of records or binary bytecode for smart contracts
nonce	the serial number of transactions

In this work, the additional data field is used to store the hash of records. Figure 1 shows the detailed logical flow for processing a transaction. In detail, the main steps of this interaction process can be described as follows:

- All nodes should run on the P2P network. When a node sends a transaction to the blockchain, all of the member nodes will receive the transaction, and this is called the broadcast process of the transaction.
- All the nodes receiving the transaction need to check this transaction and determine whether it is a legal transaction, then the legal transaction will be put into the transactions pool.
- All of the miner nodes compete for the accounting right of the transaction based on the consensus algorithm.
- After the transaction is packaged into a block by the selected miner node, the block copy is sent in broadcasted to the whole network, then nodes verify whether the block is legal or not.

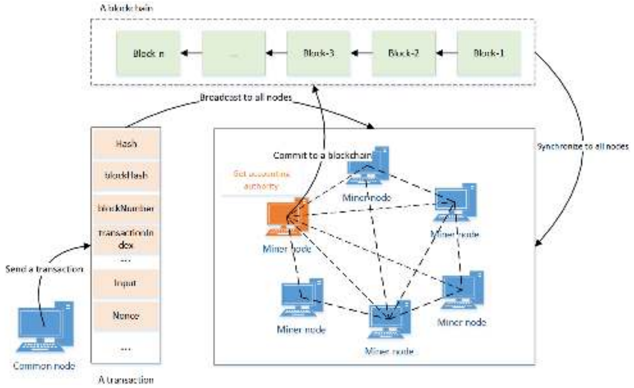


FIGURE 1. The detailed logical flow for processing a transaction.

- When the block is confirmed, then the transaction will be included in the blockchain, and all nodes in blockchain will save the backup of the current blockchain.

B. DISTRIBUTED CONSENSUS

The blockchain is based on the P2P network where each network node will undertake the tasks of network routing, transaction verification, the discovery of new nodes and so on. The miner nodes in the blockchain need to compete for accounting authority for transactions, which is dependent on the consensus algorithm employed by the blockchain. Several common consensus algorithms have been proposed:

- Proof-of-Work (PoW): The PoW algorithm is first used in Bitcoin, which selects an accounting node by the competition of computing power. Nodes based on their computing power to compete with each other to solve a complex but easy to verify SHA256 mathematical problem, the node which first solves the problem will get the accounting authority. However, a lot of mathematical computing will lead to high energy-consuming.
- Proof-of-Stake(PoS): To solve the problem of energy-consuming in PoW. The PoS algorithm proposes to obtain accounting authority according to the asset of nodes rather than the computing power. For example, the Peercoin system adopts a consensus algorithm based on coinage, and nodes with long coinage will be more likely to obtain the accounting authority.
- Practical Byzantine Fault Tolerance (PBFT): It is a fault-tolerant technique in the distributed system. The process of PBFT mainly consists of three stages: pre-prepare, prepare and commit. The algorithm can reach consensus quickly, but this algorithm had better not to employed by the public blockchain because more nodes lead to more communication delay. So, it is usually suitable for the consortium blockchain, such as Hyperledger Fabric, due to the relatively small number of nodes.

C. SMART CONTRACTS

The smart contract is the digital form of the traditional contract and defines the rules that the participants should follow. As a concept, the smart contract has been proposed for more than 20 years, until the blockchain emerges, providing an environment for the promotion of smart contracts. Most of the current blockchain systems, including Bitcoin, Ethereum, and Hyperledger Fabric, support smart contracts. From the perspective of the runtime environment, smart contracts can be divided into *Script type* and *Turing Complete type*.

- The Bitcoin system allows for the definition of simple transaction logic by writing stack-based opcodes, known as bitcoin scripts.
- Ethereum provides a smart contract platform based on Turing Complete language (Solidity, Serpent) and it is the first Turing Complete smart contract. Ethereum system provides Ethereum Virtual Machine (EVM).The contract runs inside the EVM. Ethereum users write smart contract code in a specific language and compile it into EVM bytecode to run.
- Hyperledger Fabric provides another Turing Complete smart contract, a language-independent smart contract, in which smart contract code can be written in any programming language, is then compiled by the compiler and packaged into a Docker image, using the container as the runtime environment. Turing Complete means that any operational logic can be implemented.

In this work, the used smart contracts are all Turing Complete-based, the smart contracts provide the possibility to define complex logic on the blockchain. While the blockchain environment ensures the authenticity and uniqueness of the results of the smart contract execution.

IV. SYSTEM MODEL

A. SYSTEM ARCHITECTURE

The proposed scheme based on blockchain provides efficient records storage, sharing, and certification, which is composed of education institutions, consortium blockchain, storage server, and the framework service.

Education Institutions: As the entities corresponding to the blockchain member nodes, educational institutions are the main body of data storage and sharing on the blockchain.

Consortium Blockchain: The used consortium blockchain is responsible for storing the summary information of encrypted records. Again, Ethereum is adopted to implement the consortium blockchain. The smart contracts are deployed to control the storage and retrieval of data.

Storage Server: The original records and files are encrypted and stored in the storage server. The MongoDB database is chosen as the storage server to efficiently store and retrieve data and support encrypted storage of files.

The Framework Service: Contains a set of available functional modules, does not store any recorded information and provides services to nodes or users in the form of RESTful APIs.

The proposed scheme can be deeply integrated into the storage and management of educational records. An architectural overview of the proposed scheme can be found in Figure 2.

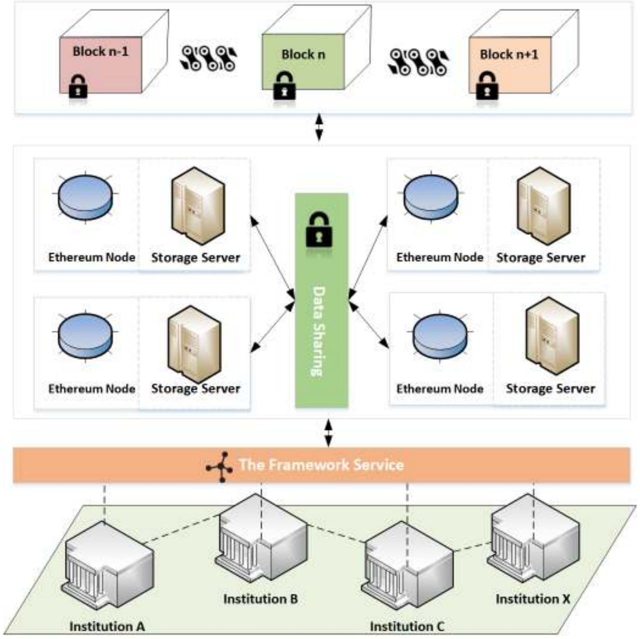


FIGURE 2. Overview of the architecture of the EduRSS.

B. BLOCKCHAIN NETWORK FOR STORAGE AND SHARING

In this work, we use Ethereum to implement the consortium blockchain, which includes $N \subseteq \{1, 2, 3, \ldots, n\}$ member nodes for records storage and sharing. These nodes can be divided into two categories.

- *Full Nodes:* The so-called full node is actually a node that synchronizes all blockchain data, including various block bodies, transaction lists and other related information.These nodes require higher computing and storage capabilities for the workstations.
- *Light Nodes:* The light nodes do not participate in downloading and maintaining the blockchain general ledger, but only maintain the block headers of the current blockchain and such nodes usually do not participate in the competition for accounting rights. Furthermore, such nodes have low computation and storage requirements for the workstations, and they are more suitable for mobile terminals and embedded devices.

Note that each node will save the view of the general ledger. They are responsible for receiving and broadcasting the transaction requests.

C. CERTIFICATE AUTHORITY (CA)

Any education institution that attempts to join the blockchain network has to set up a blockchain node. In order to maintain the security of the network, an identity authentication mechanism is introduced into the scheme. More precisely, each institution registers to the CA and obtains its public and private keys. Besides, the official information of institutions will be stored in the CA. Assuming that there could be some malicious nodes in the system, if a node files a reasonable complaint about a malicious node, then the CA could disclose the identity of the malicious node and sanctions this node. This role of the CA does not conflict with the blockchain because it serves as a key initializer and a mediation for the interaction between the external node and the blockchain member nodes.

D. BASIC ALGORITHMS

The phases of the proposed scheme are implemented through the use of several algorithms. In this subsection, we make a rough description of these algorithms:

$\boldsymbol{Setup}(1^k, U_{(X)}) \rightarrow (PK_{(X)}, SK_{(X)})$: This algorithm is used to generate the public and private keys of institutions. In detail, the RSA Or ECC algorithm is used to carry out this process. Such an algorithm is executed by the institution itself and it takes as input the security parameters k and the universal set U of attributes as input. ***Setup*** algorithm returns as output the pair of keys $PK_{(X)}$ and $SK_{(X)}$, where the former represents the public key of the record owner, whereas the latter denotes the private key of the record owner.

$\boldsymbol{Encrypt}(SK_{(X)}, PT_{(P)}) \rightarrow CT_{(P)}$: This encryption algorithm is used to encrypt the plain text of the record. Using the record owner's private key $SK_{(X)}$ as the encryption key, the method is often used to encrypt the educational records. $PT_{(P)}$ represents the plaintext characterizing an educational record, while $CT_{(P)}$ represents the ciphertext returned as output by the encryption algorithm and P refers to an educational record.

$\boldsymbol{Decrypt}(CT_{(P)}, PK_{(X)}) \rightarrow PT_{(P)}$: This decryption algorithm is usually executed by member nodes on the blockchain. $CT_{(P)}$ represents the ciphertext of P received by the member node, $PK_{(X)}$ represents the public key of the record owner. To carry out the decryption process, the receiving node of $CT_{(P)}$ needs to keep the public key of the record owner node. After the decryption, the plain text $PT_{(P)}$ will be obtained.

$\boldsymbol{calculateHash}(P) \rightarrow S_{(P)}$: This algorithm is used to calculate the hash value of record P. SHA-3 hash function family can be used to obtain summaries of records in this process. This algorithm is executed by the record owner node. $S_{(P)}$ represents the final hash value.

$\boldsymbol{sendtoCA}(CT_{(P)}, PK_{(X)}, CON_{(CA)}) \rightarrow AuthInfo_{(X)}$: This algorithm will be deployed on the blockchain so as to send the authentication information of $EDI_{(X)}$ to the CA node.

$\boldsymbol{sendTransaction}(data) \rightarrow TxId$: This algorithm is used as one of the interaction approaches between applications and blockchain. It can be called by member nodes to store records in the form of transactions. $TxId$ represents the ID of the transaction that stores the $data$.

$\boldsymbol{getTransaction}(TxId) \rightarrow TX$: This algorithm is executed by member nodes for only member nodes that have permissions to access the blockchain. $TxId$ represents the transaction ID to obtain the corresponding transaction object TX.

$\boldsymbol{VoteforJoin}(OF_{(X)}, AgreeOrNot)$: A node needs to obtain the voting approval of other member nodes in the process of joining the blockchain network. This algorithm provides a voting method for the joining process. $OF_{(X)}$ rep-

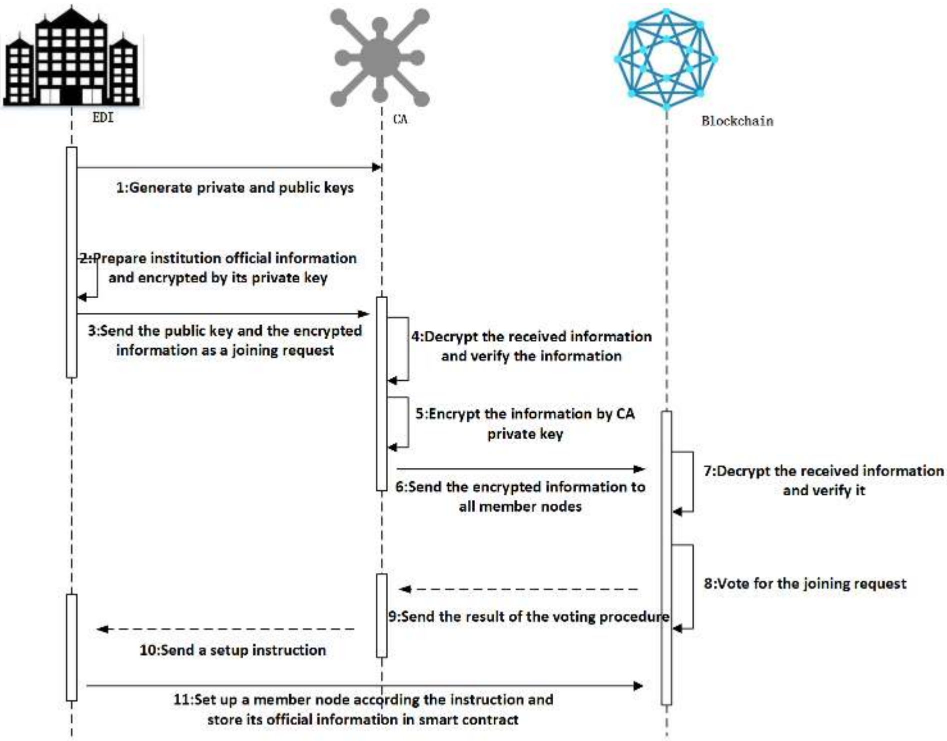

EDI
CA
Blockchain
1:Generate private and public keys
2:Prepare institution official information and encrypted by its private key
3:Send the public key and the encrypted information as a joining request
4:Decrypt the received information and verify the information
5:Encrypt the information by CA private key
6:Send the encrypted information to all member nodes
7:Decrypt the received information and verify it
8:Vote for the joining request
9:Send the result of the voting procedure
10:Send a setup instruction
11:Set up a member node according the instruction and store its official information in smart contract

FIGURE 3. The flow chart of the institutions joining process.

resents the institution's official information, $OF_{(X)}$ will be reviewed by member nodes on the blockchain. The input parameter $AgreeOrNot$ represents its attitude to the $OF_{(X)}$, the value can be either true or false, denoting "in favor" or "against", respectively. The voting information is encrypted by the $PK_{(CA)}$, and this algorithm is provided by the smart contract.

$\boldsymbol{GetVoteResult}(OF_{(X)}, SK_{(CA)}) \rightarrow VRT_{(X)}$: This algorithm is executed by the CA node to obtain the voting results of $OF_{(X)}$, $SK_{(CA)}$ represents the private key of the CA node and it is used to decrypt the voting results. $VRT_{(X)}$ refers to the final results of the voting process. This algorithm determines whether a node can join the blockchain.

$\boldsymbol{SetupMemNodes}(CERT_{(X)})$: When the $EDI_{(X)}$ obtains the certificate $CERT_{(X)}$, this algorithm is invoked to implement the settings of member nodes.

$\boldsymbol{Signature}(P) \rightarrow Sig_{(P)}$: This algorithm is used to generate a digital signature, which is usually used for messages authentication.

V. PROPOSED SCHEME

A. INSTITUTIONS JOINING PROCESS

Any new institution attempts to join the blockchain network using the framework service to generate its public and private keys. In this process, the CA is responsible for mediation during the interaction between the external node and the blockchain member nodes. Figure 3 shows the process with detailed steps as follows.

Phase1: Request to join the network.

Step1: $EDI_{(X)}$ sends its joining request to the CA, then the CA uses the setup algorithm to generate public and private keys: $\boldsymbol{Setup}(1^k, U_{(X)}) \rightarrow (PK_{(X)}, SK_{(X)})$.

Step2: $EDI_{(X)}$ encrypts its $OF_{(X)}$:

$\boldsymbol{Encrypt}(SK_{(X)}, OF_{(X)}) \rightarrow CT_{(OF_{(X)})}$ and sends the encrypted information $CT_{(OF_{(X)})}$ to the CA:

$\boldsymbol{sendtoCA}(CT_{(OF_{(X)})}, PK_{(X)}, CON_{(CA)}) \rightarrow AuthInfo_{(X)}$.

Step3: After the CA node has decrypted the $CT_{(OF_{(X)})}$ by the $PK_{(X)}$, the CA node verifies the $AuthInfo_{(X)}$. Then the CA uses its own private key $SK_{(CA)}$ to encrypt the obtained $OF_{(X)}$:

$\boldsymbol{Encrypt}(SK_{(CA)}, OF_{(X)}) \rightarrow CT_{(OF_{(X)})}$. Finally, the $CT_{(OF_{(X)})}$ will be distributed to member nodes of the consortium blockchain.

Phase2: Vote for the joining request.

Step1: When member nodes of the blockchain have received the $CT_{(OF_{(X)})}$, they use the $PK_{(CA)}$ to decrypt the $CT_{(OF_{(X)})}$ and obtain a copy of the $OF_{(X)}$.

Step2: Each member will independently verify the received the $CT_{(OF_{(X)})}$, especially officially certified agency certificates, organization codes, etc.

Step3: Each member votes on the join request, calls the deployed smart contract $IISC$(Institution Information Smart Contract), and run the ABI: $\boldsymbol{VoteforJoin}(OF_{(X)}, AgreeOrNot)$.

Phase3: Set up member nodes.

Step1:The result of voting $VRT_{(X)}$ will be obtained by running the deployed contract: $\boldsymbol{GetVoteResult}(OF_{(X)}, SK_{(CA)}) \rightarrow VRT_{(X)}$. Then the CA determines whether the $EDI_{(X)}$ can join the blockchain network based on the voting result $VRT_{(X)}$. Specifically, for the $EDI_{(X)}$, only more than two-thirds of the member institutions are in favor of its joining request can it be allowed to join the network.

Step2: The CA can only send a setup instruction $CERT_{(X)}$ to (we call it certificate) to the $EDI_{(X)}$. The $CERT_{(X)}$ should include the information necessary to set up a member node, such as genesis block configurations (like genesis.json and the networkid).

Step3: The $EDI_{(X)}$ completes the setting of the member node according to the $CERT_{(X)}$: ***SetupMemNodes***$(CERT_{(X)})$ and then invokes the ***initInstitutionMap*** of the *IISC* contract to register its official information $OF_{(X)}$.

After completing the above three phases, then the $EDI_{(X)}$ becomes a legitimate member node. To ensure the security, only member nodes can use the services provided by the model, such as saving and sharing records.

1) SMART CONTRACT DESIGN FOR IISC

IISC smart contract is employed in the joining process, the smart contract is compiled and deployed on the blockchain to control the behaviors of nodes.

IISC construction: Some state variables and a structure are defined when the contract is deployed.

- The public key of the institution $PK_{(X)}$ acts as the identity of the institution and it is recorded as *PublicKey*, *PublicKey* is then used as the index of the institution information.
- When a member node is successfully set up, a unique blockchain account address will be assigned, this address is used as the communication address between member nodes and it is recorded as *BlockAddress*. Besides, all account addresses that call smart contracts are recorded as *msg.sender*.
- *Mapping* is a map storage structure of key-value pairs used in the smart contract. The mapping relationship between *PublicKey* and institution information should be stored. Therefore, we define a mapping type variable, named *PubKeyInstitutionInfoMap* to record the relationship.
- The contract defines a structure, named *InstitutionInfo* and the *InstitutionInfo* is used to store institution-related information, such as institution certificate and organization code.

There are two main functional interfaces in the *IISC* contract listed as follows:

initInstitutionMap(*InstitutionInfo*, *PublicKey*): This function describes the process of establishing the mapping relationship between the *PublicKey* and the *InstitutionInfo*. It can only be executed by the member nodes to initialize its official information and store the information by the *IISC* contract.

isExistInstitution(*PublicKey*): This function is called to check if the specific institution's information exists on the blockchain. The input *PublicKey* is the identity of the institution and the output is boolean. In fact, we can check whether the institution exists on the blockchain by checking whether its *PublicKey* exists in the *PublicKeyInstitutionInfo Map*.

```
Algorithm 1 initInstitutionMap (InsitutionInfo, PublicKey)
Input: InsitutionInfo is the object of institution information, PublicKey is the public key of an institution.
Output: boolean.
if msg.sender is not the BlockAddress of the institution then
    return false;
end if
$PubKeyInstitutionInfoMap \leftarrow InstitutionInfo$
return true;
```

```
Algorithm 2 isExistInstitution (PublicKey)
Input: PublicKey
Output: boolean.
if PublicKey is NULL then
    return false;
end if
For(skey in keys(PubKeyInstitutionInfoMap)) do
    if $PublicKey == skey$ then
        return true;
    end if
End For
return false;
```

B. BLOCKCHAIN-ENABLED SECURE DATA STORAGE

The storage process is based on the blockchain techniques and storage servers. The transaction object on the blockchain is used to store the summaries of records, while the original records and files will be stored in the storage server after being encrypted. The decentralized environment provided by the blockchain ensures that stored summaries are not illegally tampered with. The original records stored in the storage server are periodically calibrated with the data on the blockchain to ensure security. Figure 4 illustrates the storage process.

In this process, only member nodes have the ability to store records, *IISC* contract is used to authenticate the storage request. When the $EDI_{(X)}$ attempts to send a request, the definition of the request is shown in Eq. (1):

$$SR_{(X)} = \{PK_{(X)}, TimeStamp_{(now)}, Sig\} \tag{1}$$

where $PK_{(X)}$ denotes to the public key of the $EDI_{(X)}$, $TimeStamp_{(now)}$ is the current UNIX time, *Sig* denotes the digital signature of the request.

Another important thing is how to effectively control the access permissions of the stored records. A record access permissions control contract, named *RPCC*, is used to define the permissions of record, *RPCC* contract is responsible to

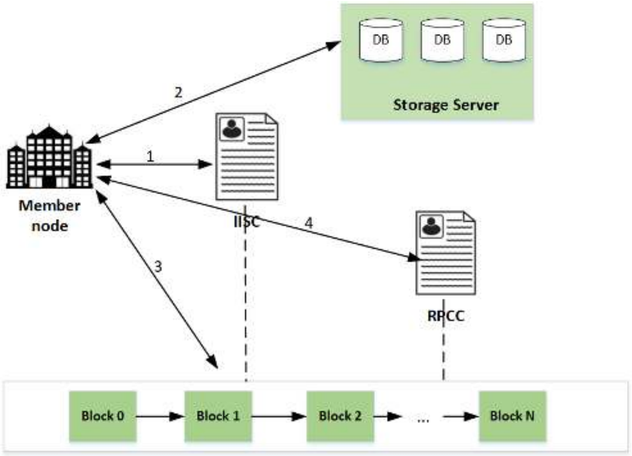


FIGURE 4. The illustration of the storage process.

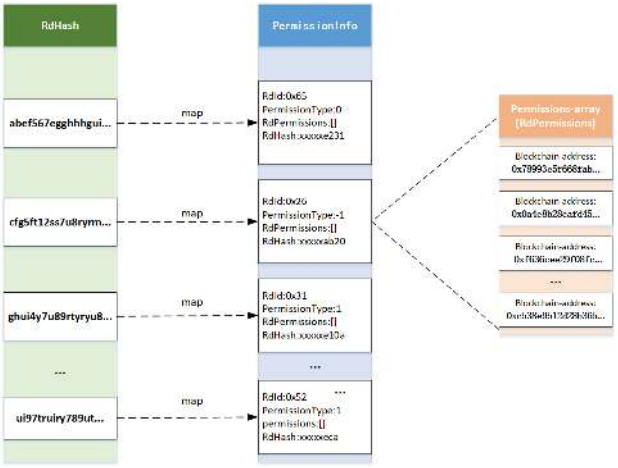


FIGURE 5. The structure of the RdPermissionMap.

establish the mapping relationship between record summary and its access permissions.

Specifically, the storage process is shown in Figure 4. In detail, the storage process works as follows:

Step1: The storage request $SR_{(X)}$ is sent from the $EDI_{(X)}$. $PK^*_{(X)}$, $TimeStamp^*_{(now)}$, and Sig^* will be extracted from the received request $SR^*_{(X)}$. $SR^*_{(X)}$ will be verified by: ***Signature***$(PK^*_{(X)}, TimeStamp^*_{(now)}) \rightarrow Sig_{(new)}$, and checking whether $Sig^* == Sig_{(new)}$. If $SR^*_{(X)}$ is a legitimate request, then the *IISC* contract will be called to identify $PK^*_{(X)}$ by: ***isExistInstitution***$(PK^*_{(X)})$.

Step2: $RD_{(X)}$ is a record that needs to be stored, first to be encrypted: ***Encrypt***$(SK_{(X)}, PT_{(RD_{(X)})}) \rightarrow CT_{(RD_{(X)})}$, $CT_{(RD_{(X)})}$ should be sent and stored in the *Storage Server*.

Step3: The summary of the $CT_{(RD_{(X)})}$ could be calculated as follows: ***calculateHash***$(CT_{(RD_{(X)})}) \rightarrow S_{(RD_{(X)})}$, and $S_{(RD_{(X)})}$ will be stored on the blockchain by sending a transaction: ***sendTransaction***$(S_{(RD_{(X)})}) \rightarrow TxId$.

Step4: The *RPCC* contract will be called by the $EDI_{(X)}$, so as to initialize the access permissions of the $RD_{(X)}$.

RPCC is deployed for setting the access permissions of records in this process. The default permissions is that no one can access the record except the record owner. *RPCC* is responsible to establish the mapping relationship between the record summary and its access permissions, the structure of this mapping is shown in Figure 5.

RPCC construction: The contract defines some state variables and a data structure when it is deployed.

The summary content $S_{(RD_{(X)})}$ is the unique identity of the $RD_{(X)}$, and it is recorded as *RdHash*. *RdHash* is the unique index of record information stored in *RPCC*.

The mapping relationship between *RdHash* and its access permissions should be stored in this contract, a mapping type variable *RdPermissionMap* is used to store such a relationship.

This contract defines a structure to store the access permissions information of record, can be recorded as *PermissionInfo*, the detail design of the *PermissionInfo* is as follows:

TABLE 3. The meanings of PermissionType.

PermissionType	Descriptions
0	No one has the access permissions besides the owner.
1	All member nodes have permissions, RdPermissions can be empty.
-1	Only the specified member nodes have permissions.

```
typedefine <Struct> PermissionInfo {
    RdId :String,
    RdHash:String,
    RdPermissions :Address[],
    PermissionType:int
 }.
```

In the *PermissionInfo* structure, an address array *RdPermissions* is defined to store the *BlockAddress* of a member node that allows access to the record. In fact, the elements in the *RdPermissions* are dynamically changed because there are cases where institutions join or exit the blockchain network. Furthermore, *PermissionType* in the structure is used to represent the permission type of record. The meanings of *PermissionType* are shown in Table 3.

RPCC provides a series of callable interfaces, the main two functions are listed as follows:

Alg. 3 shows the process of authorizing and revoking access permissions for a specified member node. Firstly, we should make sure that the *PermissionInfo* corresponding to *RdHash* already exists on the blockchain. Secondly, if the value of *Op* is equal to ‘‘add’’, then we need to clarify the record *PermissionType*, and add the input *BlockAddress* into the corresponding *RdPermissions*. Note that if the current *PermissionType* is restricted to the record owner, then granting access permissions to other member nodes is unsuccessful. Finally, if we attempt to revoke the permission of a specific *BlockAddress*, we can do this by removing the *BlockAddress* from the corresponding *RdPermissions*.

Algorithm 3 updateRdPermissions (*RdHash*, *BlockAddress*, *Op*)

```
Input: RdHash is the identity of record,
    BlockAddress is the blockchain address of the member node.
    Op is the operation type, its value can be "add" or "delete".
Output: boolean.
if RdHash does not exist OR BlockAddress does not exist then
    return false;
end if
if RdPermissionMap[RdHash] does not exist then
    return false;
end if
if Op == "add" then
    if RdPermissionMap[RdHash].PermissionType == 1 then
        return true;
    end if
    if RdPermissionMap[RdHash].PermissionType == 0 then
        return false;
    end if
    RdPermissionMap[RdHash].RdPermissions.add(BlockAddress)
    return true;
end if
if Op == "delete" then
    if BlockAddress in RdPermissions then
        RdPermissionMap[RdHash].RdPermissions.remove(BlockAddress)
        return true;
    end if
    return false;
end if
```

Algorithm 4 isExistPermission (*RdHash*, *BlockAddress*)

```
Input: RdHash as the identity of record, BlockAddress is the blockchain address of member node.
Output: boolean.
if RdHash does not exist OR BlockAddress does not exist then
    return false;
end if
if RdPermissionMap[RdHash] does not exist then
    return false;
end if
if RdPermissionMap[RdHash].PermissionType == 1 then
    return true;
end if
if RdPermissionMap[RdHash].PermissionType == 0 then
    return false;
end if
For address in RdPermissionMap[RdHash].RdPermissions do
    if address == BlockAddress then
        return true;
    end if
End For
return false;
```

Alg. 4 provides us with an approach to verify whether a member node has the access permissions on a stored record. In detail, the first thing to do is to determine whether the record information exists in the blockchain, and if *PermissionType* means that all nodes have permissions or all nodes have no permissions, it can return a certain result to the caller. Finally, if *PermissionType* cannot determine the return result, it needs to retrieve the input *BlockAddress* in the *RdPermissions*.

C. BLOCKCHAIN-ENABLED SECURE DATA SHARING

Traditionally, educational records are stored in different separate data centers of institutions, for security policies or privacy reasons, these records are difficult to share with other institutions. In some cases, when a student transfers to another institution, his/her educational records at the previous institution may be lost, in the absence of a proper data-sharing scheme. The sharing scheme of educational records proposed in this work refers to achieve a state of secure and reliable sharing of educational records, and this sharing operation usually takes place between nodes on the blockchain(legitimate member nodes). Figure 6 shows the steps involved in the sharing process. Moreover, to track the sharing status of records, a smart contract is employed, named *RSSC*. Important intermediate information in the sharing process is stored in *RSSC* to coordinate the behaviors of all involved nodes.

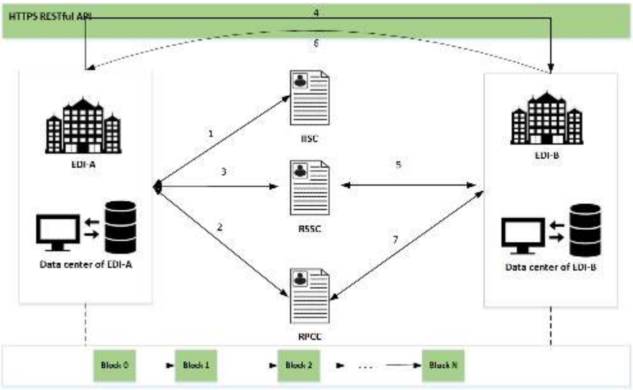


FIGURE 6. The description of sharing process.

Without loss of generality, assuming such a scenario, $EDI_{(A)}$ shares a record $RD_{(A)}$ with $EDI_{(B)}$ as described in Figure 6, the sharing process can be divided into four phases: *Initialization*, *Confirmation*, *Transmission* and *Storage*.

1) INITIALIZATION PHASE

A sharing request $SHR_{(A)}$ is sent by $EDI_{(A)}$ through the framework service. The sharing request can be defined in Eq. (2):

$$SHR_{(A)} = \{PK_{(A)}, FAddr, DAddr, TS_{(now)}, Sig\} \qquad (2)$$

where the public key $PK_{(A)}$, the blockaddress of initiator *FAddr* and the destination address *DAddr* will all be packaged into a sharing request.

The received request, referred to as $SHR^{*}_{(A)}$, then the $SHR^{*}_{(A)}$ will be parsed as follows: $\boldsymbol{UnPack}(SHR^{*}_{(A)}) \rightarrow \{PK^{*}_{(A)}, FAddr^{*}, DAddr^{*}, TS^{*}_{(now)}, Sig^{*}\}$. In detail, the $SHR^{*}_{(A)}$ will be verified as follows:

$\boldsymbol{Signature}(PK^{*}_{(A)}, FAddr^{*}, DAddr^{*}, TS^{*}_{(now)}) \rightarrow Sig_{(now)}$, and checking whether $Sig^{*} == Sig_{(now)}$. If $SHR^{*}_{(A)}$ is a legitimate request package, then $FAddr^{*}$ and $DAddr^{*}$ should be verified if they are valid blockchain addresses.

In detail, the *IISC* contract is called to verify the received addresses as follows:

$\{FAddr^{*}, DAddr^{*}\} \rightarrow IISC$, the verify interface will be invoked: $Vrfy_{addr}([FAddr*, DAddr*])$. If $Vrfy_{addr}([FAddr*, DAddr*]) == true$ then the received $FAddr^{*}$ and $DAddr^{*}$ are both valid addresses on the blockchain.

At the end of this phase, the shared information of this process should be recorded in the blockchain, and the current state should be set as "initial". The data structure that stores the shared information should be defined as follows:

typedefine <Struct> *SharingInfo* {
RdId :String,
RdHash:String,
FromAddress:Address,
DestinationAddress:Address,
SharingStatus:String,
ShareKey:String
}.

RdId is defined as the ID of the storage data used to retrieve the record in the storage servers, *FromAddress* records the *BlockAddress* of the record owner and *DestinationAddress* is the BlockAddress of the node that will receive the record during the process. While the variable *SharingStatus* records the status of the sharing process, including "initial", "confirmed", "transmit", "complete", which correspond to the four phases of the sharing process. *ShareKey* is defined as the record owner's public key and the *ShareKey* is used to decrypt the shared record.

RSSC contract provides an interface to store *SharingInfo* in the blockchain. The details are shown in Algorithm 5.

Algorithm 5 addSharingInfo (*FAddr*, *DAddr*, *Status*)

```
Input: $FAddr$ as the blockchain address of the
       record owner,
       $DAddr$ refers to the destination blockchain address,
       $Status$ refers to the current sharing status.
Output: boolean.
if $RdHash$ does not exist OR $BlockAddress$ does not exist then
    return false;
end if
if $RdSharingMap[RdHash]$ does not exist then
    $RdSharingMap[RdHash] \leftarrow ArrSharingInfo$;
end if
$SharingInfo \leftarrow InitSharingInfo(FAddr, DAddr, Status)$;
    $RdSharingMap[RdHash]$.add($SharingInfo$);
return true;
```

2) CONFIRMATION PHASE

One key issue that needs to be considered is whether $EDI_{(B)}$ has access permission to the document $RD_{(A)}$. Moreover, the *RPCC* contract can be used to confirm the access permission of $EDI_{(B)}$. To carry out this task, the $\boldsymbol{isExistPermission}(RdHash_{(RD_{(A)})}, DAddr^{*})$ is used. More precisely, if the ***isExistPermission*** runs as the output of *true*, this means that $EDI_{(B)}$ already has the permission; otherwise, the $EDI_{(A)}$ needs to confirm whether to grant permissions to $EDI_{(B)}$.

After determining that $EDI_{(B)}$ has access permission to the $RD_{(A)}$, the interface *updateSharingStatus* provided by *RSSC* should be called to update the current sharing status.

3) TRANSMISSION PHASE

The transmission phase refers to the transmission of the shared record $RD_{(A)}$, which requires the participants to collaborate. After $EDI_{(B)}$ gets the permissions of $RD_{(A)}$, the process can enter the transmission phase.

$EDI_{(A)}$ notifies $EDI_{(B)}$ to call the *RSSC* contract by broadcasting a synchronization message in a secure manner, and the synchronization is shown in Eq. (3):

$$SYN = \{RdHash, FAddr, DAddr, TS_{(now)}, Sig\} \qquad (3)$$

where $SYN_{(B)}$ should be encrypted by $PK_{(B)}$ to ensure that it can only be parsed by $EDI_{(B)}$:

$\boldsymbol{Encrypt}(PK_{(B)}, PT_{(SYN_{(B)})}) \rightarrow CT_{(SYN_{(B)})}$. Again, $Sig_{(PK_{(B)})}$ is used to verify the integrity of $SYN_{(B)}$. In detail, $Sig_{(PK_{(B)})}$ is calculated as follows:

$\boldsymbol{calculateHash}(SYN_{(B)}) \rightarrow S_{(SYN_{(B)})}$,

$\boldsymbol{Signature}(PK_{(B)}, S_{(SYN_{(B)})}) \rightarrow Sig_{(PK_{(B)})}$.

The *RSSC* contract is called by $EDI_{(B)}$, and the necessary information for the next steps will be obtained. *RdHash* and *RdId* are used to retrieve $CT_{(RD_{(A)})}$, while *ShareKey* is used to decrypt $CT_{(RD_{(A)})}$. Algorithm 6 shows how to get the *SharingInfo*.

$EDI_{(B)}$ has acquired $CT_{(RD_{(A)})}$ through a secure channel(e.g., through HTTPS), while *RdHash*, *RdId* are used as retrieval parameters. Finally, the storage server of $EDI_{(A)}$ feedbacks $CT_{(RD_{(A)})}$ to $EDI_{(B)}$.

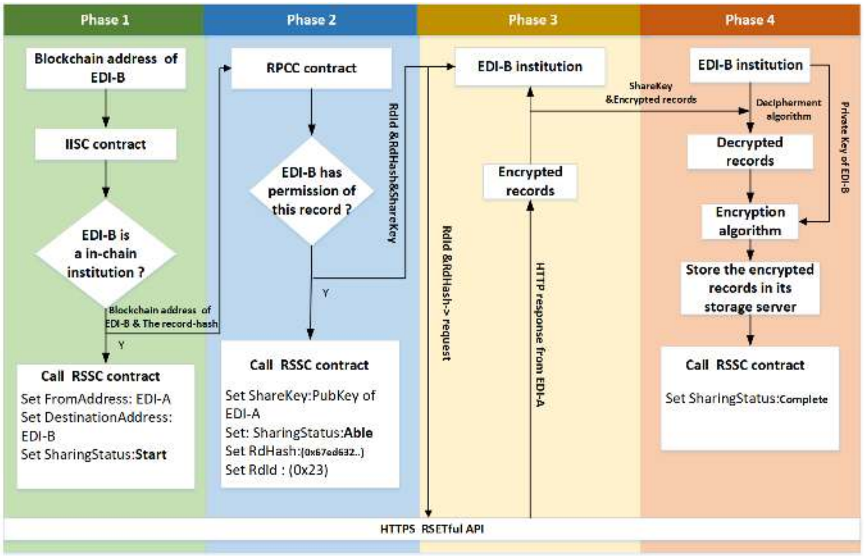


FIGURE 7. The flow chart of the sharing process.

4) STORAGE PHASE

The obtained $CT_{(RD_{(A)})}$ should be stored by $EDI_{(B)}$ in its own local *storage server*. $CT_{(RD_{(A)})}$ should be first decrypted by $PK_{(A)}$: $\boldsymbol{Decrypt}(PK_{(A)}, CT_{(RD_{(A)})}) \rightarrow PT_{(RD_{(A)})}$, then $PT_{(RD_{(A)})}$ will be encrypted by $EDI_{(B)}$:

$\boldsymbol{Encrypt}(SK_{(B)}, PT_{(RD_{(A)})}) \rightarrow CT_{(RD_{(B)})}$

$CT_{(RD_{(A)})}$ will eventually be stored in the storage server of $EDI_{(B)}$.

After completing the above steps, it means that the sharing of the record $RD_{(A)}$ has been completed, *RSSC* contract will be called, and the sharing status is updated to "complete". Figure 7 shows the detailed process of the above phases.

5) RSSC CONSTRUCTION

Some state variables of the contract and a data structure are defined in this contract.

The record hash is defined as a contract state variable, which represents the unique identity of the record to be shared and should be recorded as *RdHash*.

Since a record may contain multiple pieces of shared information, an array of the *SharingInfo* type should be defined to store all the shared information for the same record. The array is recorded as *ArrSharingInfo*.

The mapping relationship between *RdHash* and *ArrSharingInfo* represents the information about all the sharing processes of the record. *RdHash* is the index of the mapping, thus a mapping type variable *RdSharingMap* is used to store the relationship.

Several callable interfaces are provided by *RSSC*. Here, we just introduce the following main interface:

Alg. 6 provides us with an approach to get the *SharingInfo*. Considering that during the sharing process, the node of the destination address needs to obtain record sharing information belonging to it, such as hash and ID values of the shared record. In Alg. 6 we define the approach for obtaining the shared information in detail.

Algorithm 6 getSharingInfo (*RdHash*, *BlockAddress*)

```
Input: $RdHash$ is the identity of record, $BlockAddress$ is the destination address.
Output: $SharingInfo$.
if $RdHash$ does not exist OR $BlockAddress$ does not exist then
    return Null;
end if
if $RdSharingMap[RdHash]$ does not exist then
    return Null;
end if
$RdSharingMap[RdHash] \rightarrow arrSharingRd$;
For $sharingInfo$ in $arrSharingRd$ do
    if $sharingInfo.DestinationAddress == BlockAddress$ then
        return $sharingInfo$;
    end if
End for
return Null;
```

D. ANTI-TAMPERING INSPECTION BASED ON BLOCKCHAIN

There is a situation in which educational records in institutions' storage servers are tampered with and will only be found when it is proofread with the data in the blockchain. The tampered records stored in the storage server for a long time will be a security risk. In order to detect tampering in time, an anti-tampering check mechanism is introduced into the scheme, anchoring the off-chain records with their hash stored in the chain with this mechanism. The design of the

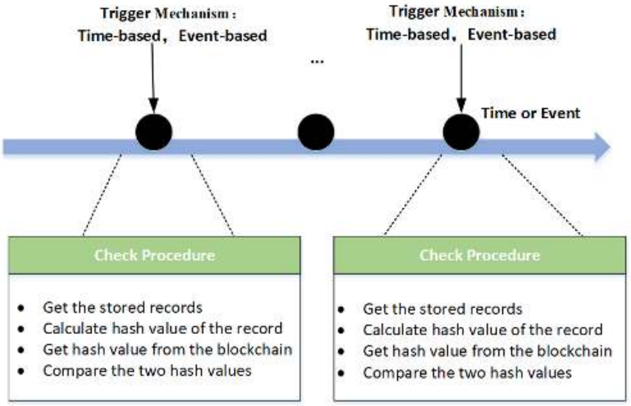

Trigger Mechanism：
Time-based，Event-based
...
Trigger Mechanism：
Time-based，Event-based
Time or Event
Check Procedure
Get the stored records
Calculate hash value of the record
Get hash value from the blockchain
Compare the two hash values
Check Procedure
Get the stored records
Calculate hash value of the record
Get hash value from the blockchain
Compare the two hash values

FIGURE 8. An overview of the anti-tampering check mechanism.

mechanism contains two aspects: the triggering of the mechanism and the design of the specific inspection procedure.

The triggering of the mechanism refers to trigger the execution inspection procedure when the defined conditions are satisfied.Time-based inspections and event-based inspections are commonly used. The time-based inspections are performed by setting a timing program that triggers the inspection procedure at a specified time point or interval. For instance, the inspection procedure can be set to be triggered every day at 19:00 or every week. While the event-based inspections trigger inspection execution based on whether a defined event has occurred. Specifically, we can define a trigger program in the storage server that triggers the inspection procedure based on the modification or deletion events. In this work, both time-based and event-based are adopted to trigger the inspection procedure. An overview of the proposed mechanism is shown in Figure 8.

In terms of the design of the inspection procedure, by comparing the hash of the current record with the hash previously stored in the blockchain, it is determined whether the processed record is legitimate, due to the non-tamperable characteristics of the blockchain. This procedure is analyzed as follows.

$CT_{(RD_{(X)})}$ as an encrypted form of $RD_{(X)}$ is stored in its *Storage Server* and it is selected as the record to be verified. The *TxId* of the transaction that stores the hash of $RD_{(X)}$ is obtained from the *Storage Server*. The $PT_{RD_{(X)}}$ will be obtained by: $\boldsymbol{Decrypt}(CT_{(RD_{(X)})}, PK_{(X)}) \rightarrow PT_{(RD_{(X)})}$, then the current hash of $RD_{(X)}$ is calculated: $\boldsymbol{calculateHash}(PT_{(RD_{(X)})}) \rightarrow S^*_{(RD_{(X)})}$, TX as the transaction object of $RD_{(X)}$ will be obtained by $\boldsymbol{getTransaction}(TxId) \rightarrow TX$. Finally, the procedure verifies the validity of $RD_{(X)}$ by checking whether $S^*_{(RD_{(X)})}$ equals to the stored hash.

Since the consortium chain has fewer nodes in the network at the beginning stage, the defense capability against the collusion attack is low. Therefore, how to adequately protect the data of the consortium chain needs to be considered. Owing to the public chain has more nodes, its decentralization is higher, and data protection is achieved by periodically anchoring the snapshot of the consortium chain to the public chain, thus we can realize a protection chain from the public chain to the consortium chain and then to the storage server.

TABLE 4. The runtime environment configurations.

OS System	Ubuntu 18.10 X64 Sever	**Hard disk capacity**	250GB
RAM	8GB	**CPU**	AMD Ryzen5
Blockchain	Ethereum	**Develop tools**	Solidity0.5.8, Truffle5.031
DB Server	MongoDB 3.4.22	**Nodes amount**	12
Encryption algorithm	RSA or ECC	**Hash algorithm**	SHA-256
HTTP Server	Nginx 1.16.1	**Service mode**	RESTful
Blockchain client	Geth	**Client libraries**	Web3.py or Web3.js

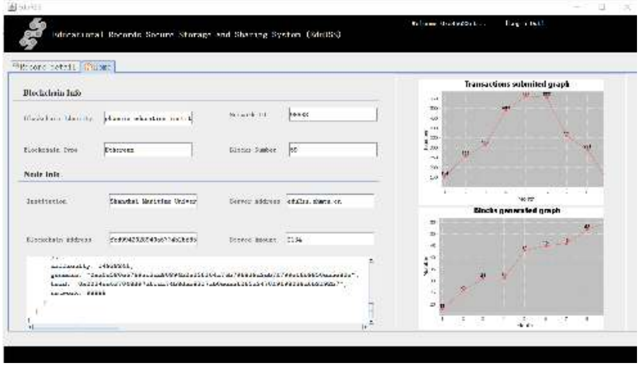


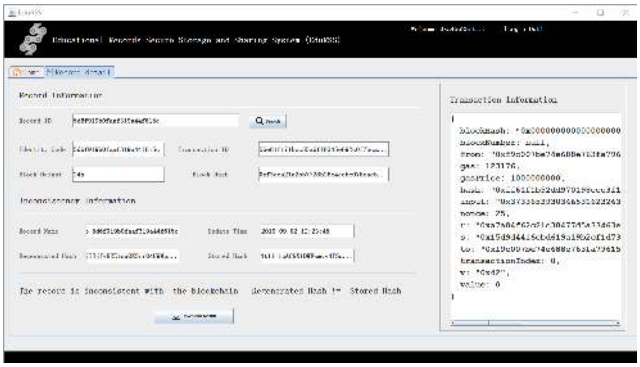
FIGURE 9. The configuration management interface.

VI. IMPLEMENTATION AND EVALUATION OF EDURSS

A. EXPERIMENTAL ENVIRONMENT AND TRIAL SYSTEM

Based on the proposed EduRSS scheme, an EduRSS trial system is developed and implemented. In detail, Alibaba Cloud platform has been chosen as our computing platform due to its excellent scalability and reliability. About 12 Ethereum nodes have been deployed to support the operation of the platform. In this trail system, Ethereum is used to provide a decentralized environment. Both the hardware and software testing environments are described in Table 4. The main GUI designs of the platform are shown in Figures 9-11.

The smart contract is an important part of the trial system. We developed the smart contracts (including *IISC*, *RPCC* and *RSSC* contracts mentioned in the above sections) by the Solidity language. Moreover, the Truffle framework is employed as a development platform for smart contracts in this work. The Truffle framework is an Ethereum-like blockchain simulation platform, we can compile and deploy the smart contracts in this platform. After completing the test of the smart contracts, we migrate these contracts to our trial system. In detail, these contracts act as interfaces for data storage and verification. Furthermore, the Ganache GUI client is used to monitor the running status of the blockchain, so we can intuitively know the running information



Educational Records Secure Storage and Sharing System (EduRSS)

Record Information

Record ID: [illegible]

Block Height: [illegible]

Inconsistency information

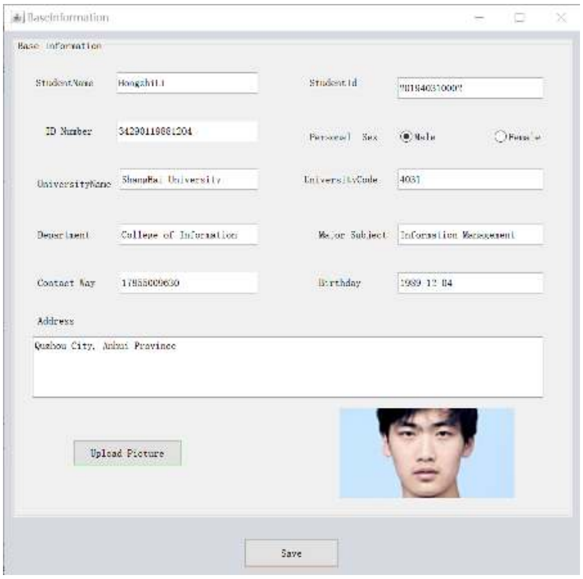
Update Time: 2019 09 02 12:25:48

The record is inconsistent with the blockchain Regenerated Hash != Stored Hash

Transaction Information

```
{
  blockHash: "0x000000000000000000
  blockNumber: null,
  from: "0x[illegible]
  gas: 123176,
  gasPrice: 1000000000,
  hash: "0x[illegible]
  input: "0x[illegible]
  nonce: 25,
  r: "0x[illegible]
  s: "0x15d9d44[illegible]
  to: "0x19e00[illegible]
  transactionIndex: 0,
  v: "0x42",
  value: 0
}
```

FIGURE 10. The detailed recording interface.



BaseInformation

Base Information

StudentName: Hongzhi Li

StudentId: "01840310007"

ID Number: 34290119881204

Personal Sex: ● Male ○ Female

UniversityName: ShangHai University

UniversityCode: 4031

Department: College of Information

Major Subject: Information Management

Contact Way: 17855009630

Birthday: 1989 12 04

Address: Quzhou City, Anhui Province

FIGURE 11. The basic user information management interface.

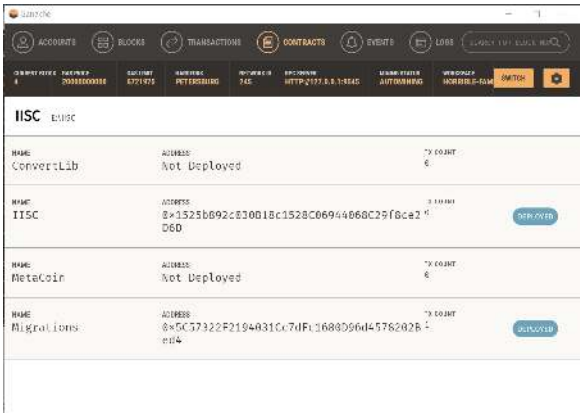
of the current blockchain from the provided GUI client (See Figures 12-13).

B. SECURITY ANALYSIS OF THE PROPOSED SCHEME

In this section, we provide a security analysis to evaluate the security and robustness of the scheme, several common attacks have been considered and the impact of each of them on the scheme is analyzed.

1) SYBIL ATTACK

Sybil attack is commonly used to attack the P2P network. In the P2P network, the same data usually needs to be backed up to multiple distributed nodes. If there is a malicious node in the network, and such node forges multiple identities through virtualization technology. As a result, the data that needs to be backed up to multiple nodes to be spoofed backup to the same malicious node (the malicious node masquerades as multiple identities). Finally, the malicious node can control the network to some extent. Since the blockchain is based on the P2P network, preventing Sybil attack is something that must be considered.

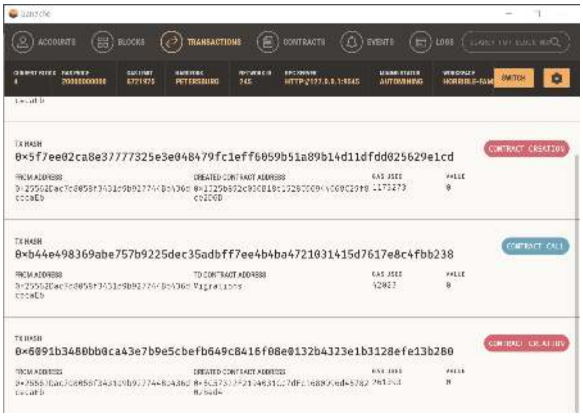


CURRENT BLOCK 4 · GAS PRICE 20000000000 · GAS LIMIT 6721975 · HARDFORK PETERSBURG · NETWORK ID 245 · RPC SERVER HTTP://127.0.0.1:8545 · MINING STATUS AUTOMINING · WORKSPACE HORRIBLE-FAM

IISC EduISC

NAME	ADDRESS	TX COUNT	
ConvertLib	Not Deployed	0	
IISC	0×1525bB92c830B18c1528C06944868C29f8ce2D6D	[illegible]	DEPLOYED
MetaCoin	Not Deployed	0	
Migrations	0×5C57322F2194031Cc7dFc1680D96d4578202Be04	1	DEPLOYED

FIGURE 12. The information of the deployed contracts.



CURRENT BLOCK 4 · GAS PRICE 20000000000 · GAS LIMIT 6721975 · HARDFORK PETERSBURG · NETWORK ID 245 · RPC SERVER HTTP://127.0.0.1:8545 · MINING STATUS AUTOMINING · WORKSPACE HORRIBLE-FAM

TX HASH 0×5f7ee02ca8e37777325e3e048479fc1eff6059b51a89b14d11dfdd025629e1cd — CONTRACT CREATION

FROM ADDRESS	CREATED CONTRACT ADDRESS	GAS USED	VALUE
0×25502Dac7c8058f3431c9b92774[illegible]cccaEb	0×1525bB92c830B18c1528C0694[illegible]C29f8ce2D6D	1173273	0

TX HASH 0×b44e498369abe757b9225dec35adbff7ee4b4ba4721031415d7617e8c4fbb238 — CONTRACT CALL

FROM ADDRESS	TO CONTRACT ADDRESS	GAS USED	VALUE
0×25502Dac7c8058f3431c9b92774[illegible]cccaEb	Migrations	42023	0

TX HASH 0×6091b3480bb0ca43e7b9e5cbefb649c8416f08e0132b4323e1b3128efe13b280 — CONTRACT CREATION

FROM ADDRESS	CREATED CONTRACT ADDRESS	GAS USED	VALUE
0×2550[illegible]cccaEb	0×5C57322F2194031Cc7dFc1680D96d4578202Be04	2613[illegible]	0

FIGURE 13. The information of the submited transactions.

However, in the proposed EduRSS scheme, if a node attempts to join the network, its official information should be provided. Such information is usually unique, contains the official authorization, and can be used to identify the node. Moreover, other nodes in the network will review the information and vote for the join request. To verify our analysis, we have selected a legitimate node in the trial system, use VirtualBox 6.0 to create a virtual computer on this selected node, and forge another identity try to join the blockchain network. But the result is a failure, because the forged identity cannot be authenticated. Therefore, it is difficult to forge multiple identities for each node, and there is no advantage to be gained via a Sybil attack.

2) HOSTAGE BYTE ATTACK

The hostage byte attack is an attack against distributed storage where malicious nodes in EduRSS refuse to transfer records or portions of records, especially during the nodes synchronization process of the blockchain, such nodes usually attempt to extort additional payments.

One of the practical approaches to defense such kind of attack is the use of redundant storage. In fact, data nodes reduce the probability of being attacked by multiple backups to other nodes. Since full nodes in the blockchain will back up all the data of the entire network, the attacked node can acquire data from other full nodes, so the number of full nodes determines the ability to resist the attack. Therefore, maintaining a certain proportion of full nodes can effectively cope with attacks initiated by some malicious nodes. Finally, with the help of the network analysis tool WireShark 3.0.1, we have verified our analysis of hostage byte attack.

3) COLLUSION TAMPER ATTACK

Once a transaction is successfully committed to the blockchain, such a transaction will be distributed and stored in each node. However, in such cases, more than half of the nodes collude to tamper with the stored transaction data at the beginning stage of the blockchain network. In fact, this attack could be achieved when the number of member nodes is small.

Therefore, the EduRSS scheme periodically anchors the data of the consortium chain to the public Ethereum. Due to a large number of nodes and blocks in the public chain, the possibility of colluding to tamper is small. The anchored data on the public chain protect the data on the consortium chain, and data on the consortium chain protect the data stored on the local data center. Hence, such a data protection chain will reduce the possibility of collusion tamper.

4) REPLAY ATTACK

Replay attack is a typical cyber attack in the P2P network. In fact, the attacker achieves identities camouflage by intercepting and replaying messages from a specific host. The consortium blockchain is based on the P2P network, each node acts as both the server and client. Therefore, it is still necessary to consider the replay attack in our proposed scheme.

Assume that there are a normal user U, a server S, and an attacker T in the network. M_u represents the plaintext of U's message. $E_{(Y_u)}(M)$ refers to the signature message of user U. $E_{(K_u)}(M)$ refers to the message encrypted by user U's private key K_u, $T(U)$ indicates that the attacker T pretends to be the user U.

The attacker T should obtain specific access permissions of the blockchain network. Without the access permissions, it can neither intercept messages of nodes nor interact with nodes on the blockchain network. Therefore, T should have the basic functions of common users, that is to say, T should maintain the public keys of other nodes and send and receive messages normally on the network as follows: $T \rightarrow S : E_{(Y_t)}(E_{(K_{(t)})}(M_t))$, $S \rightarrow T : E_{(Y_s)}(E_{(K_{(s)})}(M_s))$. If the attacker T has intercepted the message M_u: $U \rightarrow T(S) : E_{(Y_u)}(E_{(K_{(u)})}(M_u))$, in this case, a replay attack can be initiated by the attacker T, after obtaining the M_u, T pretends to be the user U, and send the M_u to the server S: $T(U) \rightarrow S : E_{(Y_u)}(E_{(K_{(u)})}(M_u))$. Finally, the attacker T receives M_s as: $S \rightarrow T(U) : E_{(Y_s)}(E_{(K_{(s)})}(M_s))$. But as mentioned above, only summary information is stored in the blockchain, so M_s contains summary information, and T can not recover the detail information through the summary information.

If the attacker T pretends to be server S, and sends a message M_t to U: $T(S) \rightarrow U : E_{(Y_t)}(E_{(K_{(t)})}(M_t))$, where T can only encrypt M with its own private key K_t. However, the user U can not decrypt the signed message using the public key of server S. Based on the above analysis, we use Ettercap 0.8.3 as the simulation tool of a replay attack. After the verification of simulation attacks, it is proved that our scheme can defend against such an attack.

5) ATTACK AGAINST THE CONSENSUS ALGORITHM

The consortium blockchain is utilized to realize decentralized storage. Usually, an attacker attacks the consensus algorithm to achieve the purpose of tampering with block data. In this work PoW as an example of analyzing potential security risks, the attack model proposed in [29] is employed to analyze the attack process. Considering such a scenario if there is a malicious node in the blockchain that modifies the data stored in a block on its node, the malicious node links the modified block to form a chain by competing for new blocks. In this way, there are two versions of the chain in the blockchain, namely, the honest chain and the modified chain. The longest chain will eventually be chosen as the main chain and the shorter chain will be discarded, that is, if the malicious node wants to tamper successfully, it must make the modified chain become the longest chain.

The process of competing length between the honest chain and the modified chain can be described using Binomial Random Walk. Assume the length between the modified chain and the honest chain differs by z blocks, the modified chain becomes the longer chain and succeeded in making up for the distance gap, then the possibility of this process can be approximately considered as Gambler's Ruin problem. The probability can be calculated as Eq. (4).

z = block distance between the honest chain and the modified chain.

p = probability the honest chain gets the next new block.

q = probability the modified chain gets the next new block.

p and q represent the probability of mutually exclusive events and p + q = 1.

q_z = probability the modified chain catch up from z blocks behind.

$$q_z = \begin{cases} 1, & p <= q \\ (q/p)^z, & p > q \end{cases} \tag{4}$$

Assuming the blockchain generates a new block per the average expected time, the extended length of the modified chain will be a Poisson distribution. We multiply the Poisson density by the probability of the modified chain catch up from. The probability can be calculated as Eq. (5).

$$p_a = \sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} * \begin{cases} (q/p)^{(z-k)}, & k \leqslant z \\ 1, & k > z \end{cases} \tag{5}$$

The simulation is carried out by Matlab. As shown in Figure 14, when the probability q of the attacking node

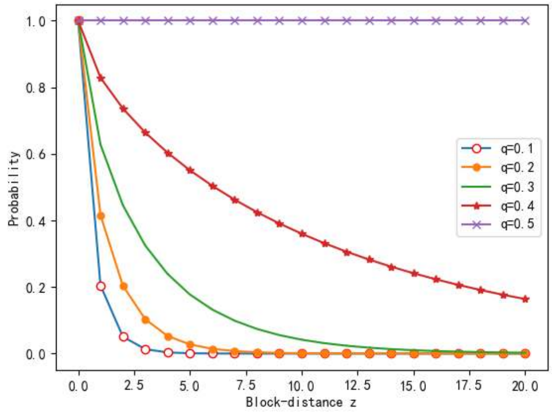


FIGURE 14. Data tampering success probability of the proposed scheme.

obtaining the accounting right is constant, with the block distance z increases, the probability p_a shows a slow downward trend. When the block gap z is the same, p_a shows a rapid upward trend with the increase of p. When p = 0.5 or more, that is to say, when the modifiers have mastered more than 50% computing power. Only on this condition, it is possible to control all the data of the entire blockchain and break through the consensus algorithm. Therefore, an appropriate increase in block distance z can defend against tampering attacks.

C. PERFORMANCE EVALUATION

1) LOAD TEST ANALYSIS

The hash information of the educational records is added to the blockchain through EduRSS. Block generation and data submission are set to asynchronous, so the consensus time between member nodes can be ignored. Besides, the educational records of different periods are selected respectively. The evaluation indicators used in this test are the minimum response time (Min), the average response time (Avg) and the maximum response time (Max). The load test is based on the Apache JMeter 5.2. We have simulated 200, 400, 600 and 800 virtual institutions by setting the corresponding number of threads. Moreover, for each thread, the number of storage requests is set to a random integer between 1 and 50.

The results are shown in Figure 15. The x-axis denotes the number of virtual institutions, while the y-axis represents the response time for the storage request. As can be observed from this figure, with the increase of the virtual institutions, the response time also increases significantly. More precisely, when the number of virtual institutions is from 200 to 800, the response time includes the minimum, the average and the maximum all increase accordingly. Overall, the maximum response time is still less than 10ms.

2) COMPARISON TO TRADITIONAL SCHEME

Ciphertext Policy Attribute-Based Encryption (CP-ABE) is a typical encryption technique in cloud storage and sharing environments. Compared with the traditional public-key encryption system (e.g., identity-based encryption IBE),

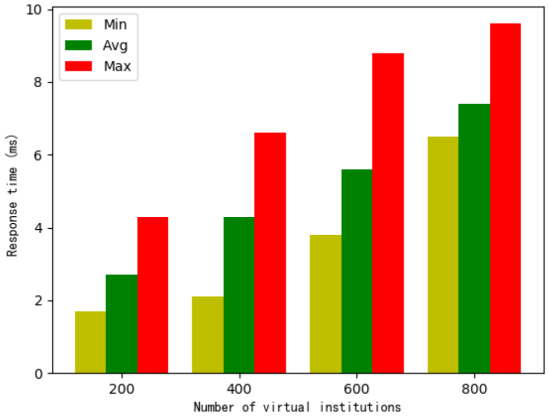


FIGURE 15. The load test results of the proposed scheme.

TABLE 5. Comparison between the CP-ABE, the MA-CPABE and the proposed scheme.

Features	CP-ABE	MA-CPABE	Our scheme
Privacy protection	Yes	Yes	Yes
Tamper resistance	Yes	Yes	Yes
Attributes based encryption	Yes	Yes	No
Control of stored records	Incomplete	Incomplete	Complete
Public and private keys pair	Incomplete	Incomplete	Complete

CP-ABE does not take the unique identification information as the user identity but uses multiple attributes (e.g., attribute sets) to identify the user, which enhances the description. CP-ABE implements access control based on the attribute granularity.

The traditional storage sharing scheme is generally based on a cloud storage solution using CP-ABE (based on single-authority) or MA-CPABE (based on multi-authority) [30], [31]. Compared with the CP-ABE and the MA-CPABE, the proposed scheme is tamper-proof and has privacy protection and secure storage. Besides, the storage scheme based on CP-ABE or MA-CPABE usually relies on a trusted third party [32], but our proposed scheme will not. Table 5 shows the comparison between the CP-ABE, the MA-CPABE and the proposed scheme.

Under the same hardware and software configuration conditions, the computational cost of the above schemes is compared. Figures 16-17 present the results.

In Figure 16, the attributes of the experimental record are limited to the same, the abscissa is the number of data packages, the ordinate is expressed as the running time of encryption and storage time. With the increase of the received data packages, the computational cost of all schemes is gradually increasing and eventually becoming stable. Overall, the computational cost of all schemes is close to each other, but when the amount of data packages increases to a certain extent, the processing efficiency of our scheme will be higher.

In Figure 17, the abscissa represents the number of

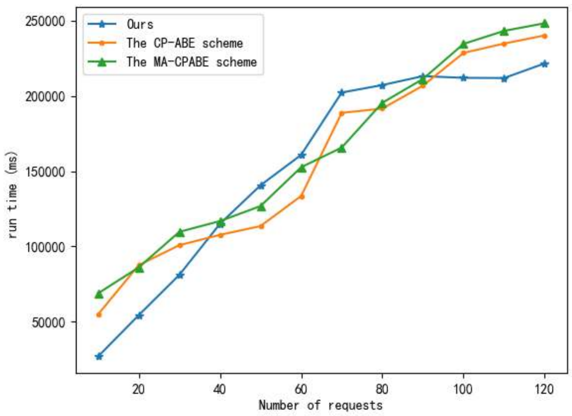


FIGURE 16. Runtime under different number of data packages.

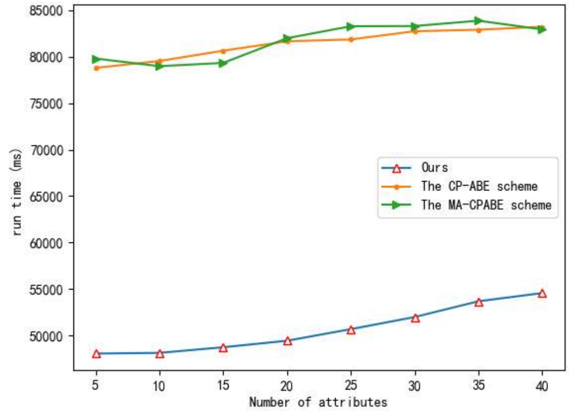


FIGURE 17. Runtime under different number of attributes.

encrypted attributes, the ordinate is the same as Figure 16 for encryption and storage time. The number of packages processed by the three schemes is limited consistent. Similarly, the run time of all schemes increases as the attributes increases. In this experiment, the CP-ABE scheme and the MA-CPABE scheme both adopt attribute-based policy encryption, in particular, the MA-CPABE is based on four attribute authorities. Our proposed scheme does not require attribute granularity encryption. As can be seen from Figure 17, the computational cost of the proposed scheme is much lower than that of the other two schemes.

VII. CONCLUSION AND FUTURE WORK

Aiming at the need for protection and sharing of educational records, a secure storage and sharing scheme based on the blockchain, referred to as EduRSS is proposed in this paper. In our proposal, the integrity and security of the data can be ensured by the consortium chain between institutions. A distributed institution authentication mechanism is proposed to ensure the security of blockchain nodes. Secure Storage is achieved by combining Blockchain and Storage Server. For records sharing, to achieve cross-institutional sharing of educational records, smart contracts are introduced, the permissions of record and the records of the sharing process are managed by smart contracts on the blockchain. Finally, an anti-tampering inspection mechanism is employed to protect records in the storage server. In theory, the proposed scheme with higher security, efficiency, and credibility, but further research works are still needed.

- We still need a secure platform to manage these smart contracts in use. Since many smart contracts have been applied in our scheme and there may be more in the future, there is a need for a professional platform for deploying, scheduling, and managing smart contracts. In addition, the security of smart contracts is one of our main focus for future research.
- More functions need to be introduced into the framework, such as support for educational record certification of external institutions or employers, and encrypted retrieval of educational records.
- The storage of the off-chain data in our scheme depends on the centralized storage server. In the future, decentralized storage technologies such as the InterPlanetary File System (IPFS) and Storj will be used.

REFERENCES

[1] *Chapman University Students' Personal Information Leaked*. Accessed: Feb. 22, 2011. [Online]. Available: https://latimesblogs.latimes.com/lanow

[2] *Leakage of Personal Information of Some Students in Changzhou University*. Accessed: Sep. 11, 2018. [Online]. Available: http://www.sohu.com/a/253122639_161795

[3] S. Mondal, K. P. Wijewardena, S. Karuppuswami, N. Kriti, D. Kumar, and P. Chahal, "Blockchain inspired RFID-based information architecture for food supply chain," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5803–5813, Jun. 2019.

[4] X. Zhang and X. Chen, "Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network," *IEEE Access*, vol. 7, pp. 58241–58254, 2019.

[5] B. Zhao, L. Fang, H. Zhang, C. Ge, W. Meng, L. Liu, and C. Su, "Y-DWMS: A digital watermark management system based on smart contracts," *Sensors*, vol. 19, no. 14, p. 3091, 2019.

[6] Z. Ma, M. Jiang, H. Gao, and Z. Wang, "Blockchain for digital rights management," *Future Gener. Comput. Syst.*, vol. 89, pp. 746–764, Dec. 2018.

[7] Y. Chen, S. Ding, Z. Xu, H. Zheng, and S. Yang, "Blockchain-based medical records secure storage and medical service framework," *J. Med. Syst.*, vol. 43, no. 1, p. 5, Nov. 2019.

[8] Z. Shae and J. J. P. Tsai, "On the design of a blockchain platform for clinical trial and precision medicine," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 1972–1980.

[9] T. M. Fernández-Caramés, I. Froiz-Míguez, O. Blanco-Novoa, and P. Fraga-Lamas, "Enabling the Internet of mobile crowdsourcing health things: A mobile fog computing, blockchain and IoT based continuous glucose monitoring system for diabetes mellitus research and care," *Sensors*, vol. 19, no. 15, p. 3319, 2019.

[10] C. Pop, M. Antal, T. Cioara, I. Anghel, D. Sera, I. Salomie, G. Raveduto, D. Ziu, V. Croce, and M. Bertoncini, "Blockchain-based scalable and tamper-evident solution for registering energy data," *Sensors*, vol. 19, no. 14, p. 3033, 2019.

[11] G. Liang, S. R. Weller, F. Luo, J. Zhao, and Z. Y. Dong, "Distributed blockchain-based data protection framework for modern power systems against cyber attacks," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 3162–3173, May 2019.

[12] M. Kim, K. Park, S. Yu, J. Lee, Y. Park, S.-W. Lee, and B. Chung, "A secure charging system for electric vehicles based on blockchain," *Sensors*, vol. 19, no. 13, p. 3028, 2019.

[13] J. Mihelj, Y. Zhang, A. Kos, and U. Sedlar, "Crowdsourced traffic event detection and source reputation assessment using smart contracts," *Sensors*, vol. 19, no. 15, p. 3267, 2019.

[14] Z. Yang, K. Zheng, K. Yang, and V. C. M. Leung, "A blockchain-based reputation system for data credibility assessment in vehicular networks," in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–5.

[15] G. Rathee, A. Sharma, R. Iqbal, M. Aloqaily, N. Jaglan, and R. Kumar, "A blockchain framework for securing connected and autonomous vehicles," *Sensors*, vol. 19, no. 14, p. 3165, 2019.

[16] A. Derhab, M. Guerroumi, A. Gumaei, L. Maglaras, M. A. Ferrag, M. Mukherjee, and F. A. Khan, "Blockchain and random subspace learning-based IDS for SDN-enabled industrial IoT security," *Sensors*, vol. 19, no. 14, p. 3119, 2019.

[17] E. Gaetani, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, and V. Sassone, "Blockchain-based database to ensure data integrity in cloud computing environments," in *Proc. Italian Conf. Cybersecur.* Venice, Italy: CINI Cybersecurity National Laboratory, Jan. 2017.

[18] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[19] Q. Xu, K. M. M. Aung, Y. Zhu, and K. L. Yong, "A blockchain-based storage system for data analytics in the Internet of Things," in *New Advances in the Internet of Things*. Cham, Switzerland: Springer, 2018, pp. 119–138.

[20] Y. Zhang, and J. Wen, "The IoT electric business model: Using blockchain technology for the Internet of Things," *Peer-to-Peer Netw. Appl.*, vol. 10, no. 4, pp. 983–994, 2017.

[21] M. Turkanović, M. Hölbl, K. Košic, M. Heri ko, and A. Kamišalić, "EduCTX: A blockchain-based higher education credit platform," *IEEE Access*, vol. 6, pp. 5112–5127, 2018.

[22] M. Sharples and J. Domingue, "The blockchain and kudos: A distributed system for educational record, reputation and reward," in *Proc. Eur. Conf. Technol. Enhanced Learn.* New York, NY, USA: Springer, 2016, pp. 490–496.

[23] G. Chen, B. Xu, M. Lu, and N.-S. Chen, "Exploring blockchain technology and its potential applications for education," *Smart Learn. Environ.*, vol. 5, no. 1, p. 1, 2018.

[24] Y. Chen, H. Xie, K. Lv, S. Wei, and C. Hu, "DEPLEST: A blockchain-based privacy-preserving distributed database toward user behaviors in social networks," *Inf. Sci.*, vol. 501, pp. 100–117, Oct. 2019.

[25] C. Knowles. *Sony Global Education Looks to Revolutionise Education With Blockchain Tech*. Accessed: Mar. 1, 2016. [Online]. Available: https://futurefive.co.nz/story/sony

[26] *BEN Provides a Community Where Students can Dream About How Blockchain can Change the World*. Accessed: May 20, 2019. [Online]. Available: https://blockchainedu.org/

[27] C. H. Liu, Q. Lin, and S. Wen, "Blockchain-enabled data collection and sharing for industrial IoT with deep reinforcement learning," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3516–3526, Jun. 2019.

[28] K. Salah, N. Nizamuddin, R. Jayaraman, and M. Omar, "Blockchain-based soybean traceability in agricultural supply chain," *IEEE Access*, vol. 7, pp. 73295–73305, 2019.

[29] *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: 2009. [Online]. Available: https://bitcoin.org/en/bitcoin-paper

[30] V. Odelu, A. K. Das, Y. S. Rao, S. Kumari, M. K. Khan, and K.-K. R. Choo, "Pairing-based CP-ABE with constant-size ciphertexts and secret keys for cloud environment," *Comput. Stand. Interfaces*, vol. 54, no. P1, pp. 3–9, Nov. 2017.

[31] Y. Xue, K. Xue, N. Gai, J. Hong, D. S. L. Wei, and P. Hong, "An attribute-based controlled collaborative access control scheme for public cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 11, pp. 2927–2942, Nov. 2019.

[32] A. Wu, Y. Zhang, X. Zheng, R. Guo, Q. Zhao, and D. Zheng, "Efficient and privacy-preserving traceable attribute-based encryption in blockchain," *Ann. Telecommun.*, vol. 74, nos. 7–8, pp. 401–411, Aug. 2019.

HONGZHI LI received the M.S. degree in computer science and engineering from the Wuhan University of Technology. He is currently pursuing the Ph.D. degree with Shanghai Maritime University. His current research interests include blockchain technology and information security.

DEZHI HAN received the B.S. degree from the Hefei University of Technology, Hefei, China, and the M.S. and Ph.D. degrees from the Huazhong University of Science and Technology, Wuhan, China. He is currently a Professor of computer science and engineering with Shanghai Maritime University. His research interests include storage architecture, cloud computing, cloud computing security, and cloud storage security technology.