Efficient Identity-based Proxy Re-encryption Scheme in Blockchain-assisted Decentralized Storage System

Jiayu He[1,2], Dong Zheng[1,2,4], Rui Guo[1,2,3], Yushuang Chen[1,2], Kemeng Li[1,2], and Xiaoling Tao[4]

(Corresponding author: Jiayu He)

School of Cyberspace Security, Xi'an University of Posts and Telecommunications[1]

National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, China[2]

Guilin University of Electronic Technology, Guilin, China[3]

Westone Cryptologic Research Center, Beijing, China[4]

Email: hjy_xupt@163.com

(Received Apr. 20, 2020; Revised and Accepted Dec. 10, 2020; First Online Aug. 14, 2021)

Abstract

The rapid development of cloud storage has greatly promoted the industrial productivity and social progress. However, with the era of big data coming, there exists several challenges for cloud storage in terms of the difficulty of maintaining data security, the efficiency of data sharing and the cost of the single point of failure. Proxy re-encryption (PRE) cryptographic primitive is regarded as a promising technique to improve the efficiency and security of data. Blockchain can overcome the flaws caused by single point of failure. These two technologies have attracted much attention recently. Nevertheless, the conventional PRE scheme incurs intricate certificate management and the blockchain is not suitable for storing big data because of the high cost. In order to figure out the above problems, in this paper, we propose a novel identity-based proxy re-encryption (IBPRE) scheme,data owner-manipulative IBPRE (DOM-IBPRE), which is achieved by combining IBPRE, blockchain and the inter planetary file system (IPFS) technology. The scheme can avoid the complexity of certificate management, enhance the security of big data storage and ameliorate the efficiency of big data sharing. In addition, we assess the security performance of the scheme by Chosen-Plaintext Attack (CPA) in the standard model and simulate our scheme with Pairing-Based Cryptography (PBC) library, by comparing with other PRE schemes, our scheme has the better performance due to the reduction of the computation times of exponential and bilinear pairings. Finally, we implement the blockchain scheme under Linux system with the hyperledger test network fabric and develop the interface to client, the test results show that our scheme is practical.

Keywords: Blockchain; Decentralized Storage System; Hyperledger Fabric; Inter Planetary File System; Proxy Re-encryption

1 Introduction

With the emergency advancement of information diversification, cloud storage is becoming increasingly popular in our daily life. Cloud storage enables enterprises or individuals to obtain cloud data at anytime and anywhere, this feature brings wonderful convenience to our lives [31]. However, there are also some challenges in existing cloud storage system. Above all, shared data is required to be encrypted to guarantee data privacy, with the era of big data coming, the difficulty of maintaining data security is getting increasingly and the efficiency of data sharing is getting reducing. In addition, many cloud storage systems are operated by a centralized corporation which has powerful ability to store and supervise data, the corporation can be seen as a third party, it ineluctable inherits the single point of failure flaws. Last but not least, with the upgrade of cloud storage devices and the rise of employee wages, the cost of centralized cloud storage is increasingly higher. Therefore, to better guarantee data privacy and availability, we should achieve a flexible access control over the encrypted data and change data storage from centralized systems to the decentralized systems [18], which not only have the cheaper price than existing centralized storage systems, but also can relieve of our concern about a single point of failure. Fortunately, the emergence of bitcoin has driven the development of blockchain technology, a blockchain can be thought of a decentralized ledger, where the data and transactions are not under the control of any third party, the data that stored on the blockchain cannot be tampered. Therefore, blockchain has strong data stability and reliability.

Inter planetary file system (IPFS) and hyperledger fab-

ric as typical distributed storage applications, they use blockchain as their core structure and attract more attentions in recent years. For the purpose of promoting efficiency and security of data encryption in cloud storage, proxy re-encryption (PRE) scheme was proposed [15], it can provide a flexible and feasible method for storing and sharing data. Nevertheless, the original PRE was proposed in the public key model which incurs intricate certificate supervision. To ease this problem, identity-based PRE (IBPRE) scheme was proposed [10], take advantage of this scheme, the identity of receivers can be seen as public keys, the process of verifying certificates is replaced by knowing the identities of users, which is more convenient in application. However, in practical applications, the master key in system is generated by single public key infrastructure (PKG), if the authenticity of the PKG cannot be trusted or the PKG is attacked maliciously, the master key may be leakage. From what we have discussed above, how to combine with IBPRE, IPFS and blockchain technology to design a flexible and efficient identity-based proxy re-encryption scheme for decentralized cloud storage is a worthy study. The specific contributions of this paper are as follows:

1) We design a frame which unites the decentralized storage system IPFS, the hyperledger fabric and IBPRE technology to achieve secure and efficient control data in decentralized system. The data owner is the sole one who can control their data, besides, a single PKG is replaced by multi trusted authority. The problem that the malicious attack on the PKG results in the leak of the private key of each user in conventional IBPRE scheme is solved.

2) We also analysis the security performance of the scheme by Chosen-Plaintext Attack (CPA) based on a modified decisional bilinear Diffie-Hellman (mDBDH) problem in the standard model. The theoretical analysis indicates that our scheme is correct with security.

3) For the purpose of mitigating the storage burden of blockchain, we only store the index of sharing files in the blockchain system. By designing the smart contract, the data user who has access rights can obtain the index of file and decrypt the file.

4) We simulate our scheme under the Linux system, the simulation of the data owner-manipulative IBPRE (DOMIBPRE) algorithm is ran based on the pairing-based cryptography library (PBC) [14], and the performance was analyzed. The implementation of the blockchain framework is performed through the hyperledger test network fabric, we design the smart contract and store the index of files to test our blockchain network, the test results prove our scheme is practical.

The remainder of paper is arranged as follows, Section 2 introduces of related work, Section 3 reviews the preliminaries, Section 4 details the system model, including scheme model and security model. The system and smart contract description are shown in Section 5. Section 6 discusses the security analysis. The performance analysis is shown in Section 7. The implementation of blockchain framework is described in Section 8. Finally, Section 9 concludes this paper.

2 Related Work

2.1 Blockchain Technology

Blockchain is a decentralized storage database, it is a novel application of computer technology such as cryptography, consensus mechanism, point-to-point transmission, smart contract and other technologies [20]. It records all transaction information which occurring on the node. All processes are highly transparent. These days, electronic cryptocurrency (such as Bitcoin [19], Ethereum [7], Zcash [5], *etc.*) have grown popular, the emergence of these cryptocurrency have promoted the development of blockchain. Taking bitcoin as an example, the transaction processes in blockchain are shown in Figure 1.

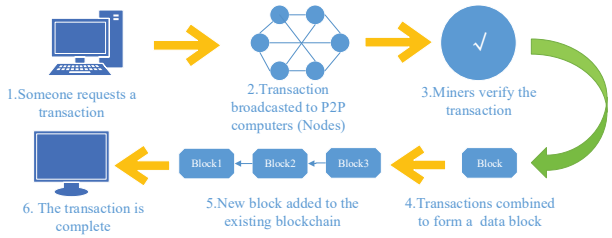

Figure 1: Transaction process of blockchain

The blockchain technology as the supporting technology of cryptocurrency has been attached more attention. With the characteristics of decentralization, transparent, anonymous traceability and non-tampering of information, it has been playing an indispensable role in many territories, such as: decentralized supply chain [1], decentralized identity-based PKI [16], decentralized IoT [17], decentralized scheme [13], decentralized storage [3], *etc.* Many researchers in cloud storage field focus on guaranteeing security of data, an individual data control system based on blockchain [34] has been proposed, this system can better guarantee the data privacy. To address the problem of the difficulty of maintaining data security and the efficiency of data sharing that impede the progress of big data, a blockchain based access control frame for reinforcing the privacy of big data platforms [27, 28] was presented. However, these schemes do not give the data owner the power to control the data efficiently and flexibly. For the purpose of achieving flexible and fine-grained data access, a scheme for cloud computing based attribute-based encryption (ABE) was proposed [22], although the scheme achieved flexible data

access, it is not effectively combined with the blockchain technology, moreover, the computation cost is large in this scheme.

Decentralized storage systems [25, 29, 33] do not rely on centralized service providers, it allows users to upload files in nodes which lease free storage space on the internet. These systems utilize blockchain as their primary structure. As a distributed storage terrace, IPFS [33] uses Filecoin as an incentive mechanism to encourage nodes to contribute retrieval service. After research, it was found that IPFS does not offer a doughty privacy encryption algorithm port for user-uploaded data. In order to solve this problem, the Storj system offers an end-to-end encryption way [29]. This system stores hash value of data on blockchain while offering a means of validating data integrity. Blockchain technology and peer to peer storage network were also employed in Sia platform [25], this platform unpicks the uploaded data into multiple data parts, and encrypts each part of data. Encrypted data are sent through smart contracts to each node that provides the storage service, the users pay siacoin for the storage service, as the storage node, they need termly submit proof of the stored data to prevent the storage node from removing the stored data. Nevertheless, in these systems, the process of data encryption is not fully controlled by the data owner.

2.2 Identity-based Proxy Re-encryption Technology

The first PRE scheme was proposed by Blaze, Bleumer and Strauss in [10], following this founding work, a series of PRE schemes were proposed in classical public key setting, such [9,12,26], these schemes require a certificate to validate the public key before encrypting a plaintext.

For the purpose of avoiding the overhead to notarize public keys certificates, Green and Ateniese [8] proposed the first IBPRE scheme, it avoids the complexity of the certificate management, after that, a number of IPRE schemes [2, 24] have been presented based the thought of identity-based encryption [23].

The above PRE schemes only allow data sharing in a rough level. In other words, the schemes could not control the process of data re-encryption. This issue is addressed in the recent conditional proxy re-encryption (CPRE) schemes [11, 32], they achieve fine-grained data sharing. In order to further improve the CPRE scheme, the conditional identity-based broadcast PRE (CIBPRE) scheme [30] was proposed, it combines the thoughts of CPRE, IPRE and broadcast encryption. In CIBPRE system, a sender can encrypt the data with the identity of receiver and data sharing conditions, the primal authorized receivers can obtain the data with their private keys, the new authorized receiver can also access the data by decrypting the ciphertext with their private keys. It avoids repeated downloading and encryption of data by the sender. These merits make CIBPRE get a practical and secure tool to store data, particularly when there are diverse receivers to share the data. The recent work in [4] proposed a more efficient conditional identity based broadcast proxy re-encryption(CIBPRE) scheme that supports diverse receivers to share the data as time passes. However, these schemes are not used decentralized storage system.

3 Preliminaries

In this part, we retrospect few of the notations and correlative background knowledge that will be needed in our paper. Table 1 shows a few of the notations deployed in the paper.

Table 1: Notations

Notations	Descriptions
MTA	Multi Trust Authority
DO	Data Owner
DU	Data User
EP	Encryption Proxy Server
Dec	Decryption Proxy Server
S	System master key
sk_{id}	The secret key of system user
pk_{id}	The public key of System user
C_{owner}	Data owner encrypted ciphertext
C_{Enc}	Encryption proxy encrypted ciphertext
C_{Dec}	Re-encryption proxy encrypted ciphertext
C'_{Dec}	Decryption Proxy encrypted ciphertext
C_T	Decryption Proxy decrypted ciphertext
tid_{An}	Transaction number
Loc	File Location

3.1 Bilinear Mapping and Computational Assumption

1) Bilinear mappin: Let G_1 and G_T be cyclic multiplicative group of big prime order q, g be a generator of G_1, We define e: $G_1 \times G_1 \rightarrow G_T$ e:is a bilinear pairing map. If e: $G_1 \times G_1 \rightarrow G_T$ has the following characters:

   Bilinear: $e(g^a, g^b) = e(g, g)^{ab}$ for all $a, b \in Z_q^*$.

   Non-degenrate: There exists $g_1, g_2 \in G_1$,such that $e(g_1, g_2) \neq 1$.

   Computable: There is an competent algorithm to compute $e(g_1, g_2)$ for all $g_1, g_2 \in G_1$.

2) Computational assumption: Let $(q, g, G_1, G_T, e) \leftarrow Setup(k)$ for a security parameter k. The modified decisional bilinear Diffie-Hellman (mDBDH) problem is to determine whether $T = e(g, g)^{b/a}$ given a tuple $(g, g^a, g^b, g^c, T) \in G_1^4 \times G_T$, where $a, b \in_R Z_q^*$.

   We set the k to be a security parameter. Generally, we think that mDBDH assumption holds in G_1,G_T,

if for any probabilistic polynomial time (PPT) algorithm A, the following formula holds:

$$\left|Pr[A(g, g^a, g^b, g^c, e(g,g)^{b/a}) = 1 | a, b \leftarrow_R z_q^*]\right| - \left|Pr[A(g, g^a, g^b, g^c, T) = 1 | a, b \leftarrow_R z_q^*]\right| \leq V(K),$$

where $V(.)$ is a negligible function, for all polynomial functions$P(.)$, we have $V(K) < 1/p(k)$.

3.2 Blockchain Technology and Hyperledger Fabric

1) Blockchain technology: Blockchain is a decentralized and distributed ledger jointly maintained by all nodes in the blockchain network. The nodes in the network keep the same ledger, there are no third party and central authority can control the entire network.

A block header and some transactions are included in each block, each block header includes of the connection pointer to the previous block header, a merkle root with a binary tree structure and a timestamp. By this way, blocks are connected together in an orderly fashion by the hash value of the header pointer. The hash algorithm guarantees that the transaction data in each block is immutable. The structure of blockchain is shown in Figure 2.

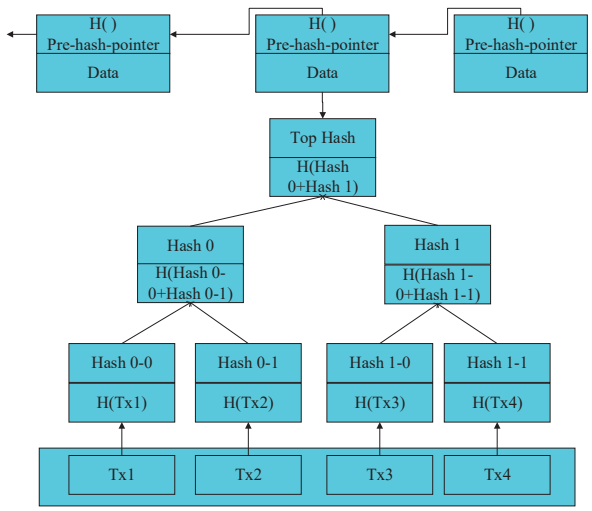

Figure 2: The structure of blockchain

In this paper, all system users are used as peer nodes to constitute a blockchain network, all of nodes are responsible for recording the index of sharing files from IPFS. There are two kinds of data in network, one is the file location, the other is a transaction number. Because the blockchain is not fit for storing big data, so we merely store the index of files in the blockchain, this way not only can alleviate the storage burden on the blockchain, but also can ameliorate the operation efficiency of the blockchain system.

2) Hyperledger fabric: Hyperledger fabric is a consortium blockchain platform and one of the hyperledger projects hosted by the Linux Foundation. It is a popular implementation of the blockchain network framework [6, 21]. As other blockchain technologies, hyperledger fabric contains a ledger, smart contracts and consensus mechanism. It is a system for managing transactions through all participants. The most obvious distinction from other blockchain systems is that fabric is private and licensed, the members of hyperledger fabric network are registered through a trusted membership service provider (MSP). Hyperledger fabric also provides the ability to create channels, permitting a group of participators to build separate transaction ledgers.

Hyperledger fabric node: Node plays a significant role in the hyperledger fabric platform, it is the core of the entire blockchain network, there are five kinds of node in fabric, CA, orderer, endorser, leader and committer. CA node is similar to a certificate authority, providing identity registration services for users. Orderer is responsible for sorting and packaging transactions into blocks on the network. Endorser is responsible for receiving a message request from client proposal, after checking the message, this node simulates the message proposal and uses its secret key to sign the simulation result, so as to show that the transaction is legal and effective, the endorser node returns the endorsement result to the client. Leader, as a representative of all nodes in the organization, it is able to connect to the orderer and broadcast the block received from orderer to all nodes in the organization. Committer is used to verify the integrity and legitimacy of the transaction message structure and to store it in the ledger.

Hyperledger fabric channel: Channel is an important concept in fabric. In essence, it is a private atomic broadcast channel divided and managed by orderer. For the purpose of isolating the information in the channel, the entities outside of the channel cannot access the information in the channel, it achieves the privacy of the transaction. At present, the channel consists of system channel and application channel. The orderer manages the application channel through the system channel, and the user's transaction information is transmitted by the application channel. For the general user, the channel is an application channel. The information in the channel is transparent to the nodes who joined the channel.

Hyperledger fabric chaincode: Smart contract is called chaincode in hyperledger fabric, the chaincode is written by developer, Go and Java language as the development languages. The

chaincode is deployed in the blockchain network node, it can run independently in a secure docker container, using the Go remote procedure calls (GRPC) protocol to correspond with related nodes and managing the data in the ledger.

To illustrate the hyperledger fabric visually, the architecture of fabric is shown in Figure 3, the complete transaction processes of fabric are shown in Figure 4.

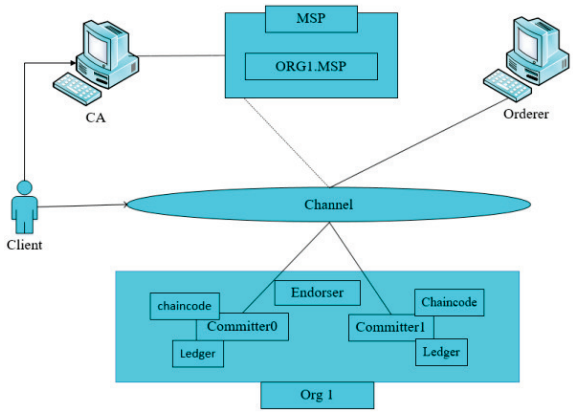

Figure 3: The architecture of fabric

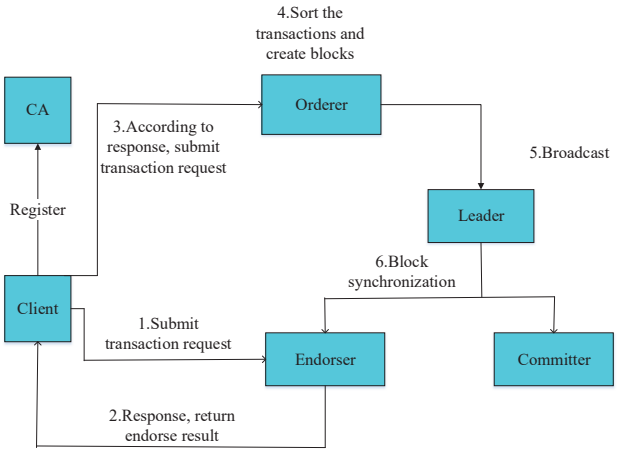

Figure 4: Transaction processes of fabric

3) IPFS: Inter planetary file system (IPFS) is peer-to-peer decentralized file system. According to the underlying protocol, the files which saved on IPFS can be obtained in anywhere. It offers a high throughput file store frame and unites techniques such as distributed hash tables (DHT), self-certifying *etc.* The merit of IPFS over extant storage is that there is no central server, therefore, this system conquers a single point of failure. When we upload the file to the system, we can obtain the hash value about file. This hash value can be seen as a Uniform Resource Locator (URL) in the web. We call this hash value file location. When the data user requires to obtain the file in IPFS, according to hash value, data user downloads the encrypted files from IPFS, and decrypted it. The complete processes of IPFS are shown in Figure 5.

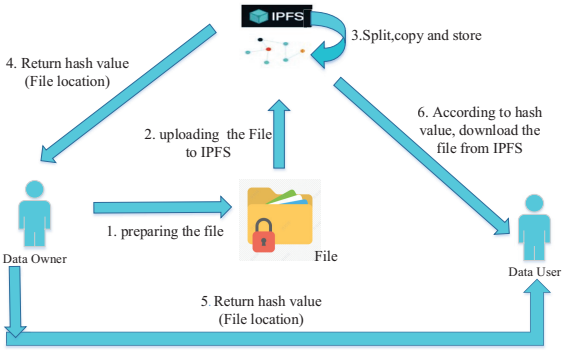

Figure 5: Transaction processes of IPFS

4 System Model

4.1 System Model

The scheme consists of the following eight entities:

Multi Trust Authority (MTA): MTA is an organization in system that cooperates to generate the master key for each user.

Data Owner (DO): DO is a person or organization that has some of files to share. He or She encrypted the file and sent it to IPFS, when received a file request from DU, DO generates a re-encryption key and sends it to the proxy, then returns the index address of the file to DU. They also write smart contracts in the blockchain to control access permissions.

Data Users:(DU): DU is the data client of DO that they are authorized to access some of sharing files. When they require to view some of the file, they send request to DO. With the help of Dec, DU decrypted the file which is shared in IPFS.

Encryption Proxy (EP): EP is a proxy that helps the DO encrypts the file again. This encryption makes the data more secure. On the basis of encryption, some ciphertext conversion work is also done for following decryption.

Inter Planetary File System (IPFS): IPFS is a peer-to-peer distributed file system. They are responsible for splitting the data into blocks and encrypting each block. Then, IPFS returns the encrypted hash value as an index to the DO.

Decryption Proxy (Dec): Dec is a proxy that helps the DO transfer the ciphertext.

They also have a responsibility to help DU decrypt the transformed ciphertext partially.

Smart contact: Smart contact is a self-executing protocol that help people to disseminate, validate, or implement a contract in an automatic manner. In our scheme, smart contract is used to authenticate the identity of user and help them upload the file index.

Blockchain:We choose hyperledger fabric as our blockchain platform. It is one of a consortium blockchain that hosted by the Linux foundation. The file index and the transaction number are stored in this blockchain. The smart contact is used on this blockchain.

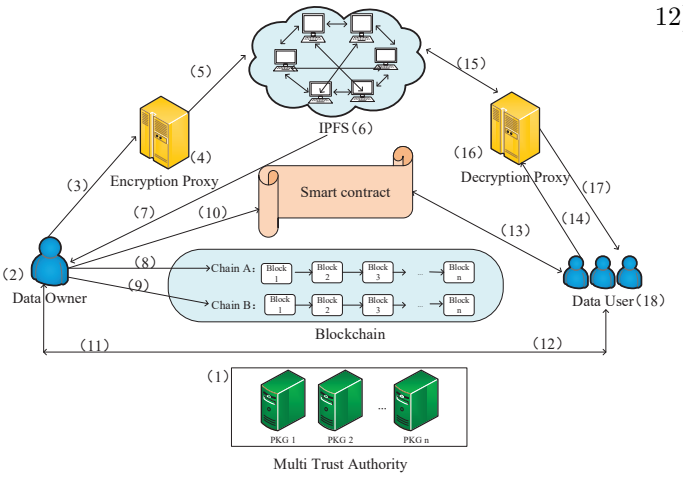

Figure 6: System framework

For clarity, the system frame is described in Figure 6. The corresponding description of each step number in Figure 6. is shown as follows:

1) Users (Including DO, DU, Enc-proxy, Dec-proxy) register in the MTA (Multi Trust Authority) and hyperledger fabric ca with their own ID, and obtain their unique public and private key.

2) DO encrypts the plain-text message m to obtain the ciphertext C_{owner}.

3) DO uploads the C_{owner} to the EP.

4) EP re-encrypts the ciphertext C_{owner} and obtains the ciphertext C_{Enc}.

5) The EP uploads the ciphertext C_{Enc} to IPFS.

6) IPFS divides the ciphertext C_{Enc} into n blocks and calculates the hash of each block. Eg: Divided C_{Enc} into n blocks: calculate $hash1 = hash(Block1)...hashn = hash(Blockn)$, the array $Arr[n] = [hash1, hash2, ..., hashn], hash(file) = hash(Arr[n])$.

7) The IPFS packages hash(file) and Arr[n] as the Loc and returns Loc to DO.

8) DO uploads Loc to chain A and records transaction number tid_{An}.

9) DO uploads the transaction number tid_{An} to chain B.

10) DO writes a smart contract, making chain B is only accessed by the DO, when the DU node queries the transaction content corresponding to the transaction tid_{An}, after confirming the identity of user, the content about tid_{An} can be returned to the DU.

11) The DU sends his/her own ID to the DO and requests to obtain the data of the DO.

12) With the help of CA and MSP, DO verifies the identity of DU, when it passes validation, DO obtains the transaction number tid_{An} from the chain B, and runs the Transkey algorithm to generate TransKey:

    a. Firstly, DO chooses the secret parameter t and computes $C_1 = g^t$.

    b. Secondly, DO chooses the secret parameter k and a hash function H_1.

    c. Thirdly, according to the identity of DU and DO, DO computes Transkey: $< R_1, R_2 >=< C_1^{k/t}, H_1(id_{DO})^{-t} \cdot H_1(id_{DU})^k >$

    d. Finally, DO returns the Transkey to DU through the fabric network. (In our scheme, we supposed that fabric network is a safe channel).

13) After verifying the identity of DU, DU gets the content from the chain A by the transaction number tid_{An}, the content is the file index Loc.

14) DU sends $(Loc, TransKey)$to the Dec.

15) Dec downloads ciphertext $CEnc$ from the IPFS according to Loc.

16) Dec converts C_{Enc} to C_{Dec} according to the Rekey, and C_{Dec} to C'_{Dec}.

17) Dec Returns C'_{Dec} to DU.

18) DU decrypts C'_{Dec} with its own private key sk_{Du}, obtains m.

In our paper, the system model consists of six parts as following:

1) $par \leftarrow setup(k)$: This step completes the system initialization setup. It takes as input security parameter k, the system run the setup algorithm, outputs a series of public parameters.

2) $(pki, ski) \leftarrow keyGen(id)$: Users in this system register in the MTA with his or her own ID, according to each ID, MTA runs algorithm to generate public and private key for each user. As shown in Step (1) of Figure 6.

3) The encryption by Data Owner: In this part, it is made up of the following two sub-algorithms: The encrypt and TranskeyGen algorithm are run by DO.

   a. $Encrypt.owner \leftarrow (ski, pki, m)$: The file encrypt algorithm takes as input the shared file m. DO chooses two secret parameters and computes, it outputs file ciphertest $C_{owner.}$ DO uploads the ciphertext C_{owner} to EP. As shown in Steps (2) and (3) of Figure 6.

   b. $TransKey \leftarrow TranskeyGen(id_{DO}, id_{DU})$: Data Owner chooses a secret parameter, receive ID from DU, then, DO runs the TranskeyGen algorithm to generate TransKey and sends it to DU. As shown in Steps (11) and (12) of Figure 6. The detailed process of TransKey algorithm is shown in the part 5.1.3).

4) The computation by Encryption Proxy: This step consists of two algorithms: RekeyGen and Re-Encrypt algorithm. They are all run by EP.

   a. $Rekey \leftarrow RekeyGen(pk_{DO}, pk_{Enc}, pk_{Dec})$: EP received three public key of the Data Owner DO, the delegator EP and the delegatee Dec. EP runs Rekey algorithm to generate Rekey and send it to Dec.

   b. $ReEncrypt.Enc \leftarrow C_{owner}$: EP received the ciphertext C_{owner} from DO, EP runs encrypt algorithm to generate ciphertext C_{Enc}, EP uploads C_{Enc} to IPFS, IPFS distributes the ciphertext and storage, then returns the index address Loc of the file to the DO. As shown in Steps (4) $\sim$ (7) of Figure 6.

5) The computation by Decryption Proxy: In this part, it consists of the following two sub-algorithms, ReEncrypt and Decrypt algorithm, both algorithms run on the Dec. Firstly, DO stores some index information into the blockchain and writes smart contracts, returns some correlative information to DU, DU sends the returned information to Dec. Secondly, according to the index information, Dec downloads the relevant files from IPFS and decrypts them. The purpose of the ReEncrypt process is to convert the ciphertext C_{Enc} into the ciphertext C_{Dec} that DU can decrypt it. In order to reduce the computation of DU, decrypt algorithm is used for partial decryption of ciphertext. This step decrypts C_{Dec} to ciphertext C'_{Dec}, then, Dec sends the ciphertext C'_{Dec} to DU. As shown in Steps (8) $\sim$ (17) of Figure 6.

6) $Decrypt.user \leftarrow (c'_{Dec})$:DU received the (c'_{Dec}), then decrypts it by his or her own secret key sk_{DU}. DU computes a bilinear pair operation and xor to get the plaintext message m. As shown in Step (18) of Figure 6.

4.2 Security Model

We supposed that the encryption proxy and the decryption proxy server are curious but honest in our system model, supposed that $\varepsilon = (Setup, KeyGen, Enc, ReKeyGen, ReEnc, Dec)$ is a DOMIBPRE scheme. In order to describe our game, denoted by $Exp_{\varepsilon,A}^{IND-DOMIBPRE-CPA}$, we devise the oracle between a PPT adversary Adv and a challenger C, that the adversary Adv can query during the game:

1) $G_{pk}(id)$: The oracle that generates the public key. Give as input id, C operates $keyGen(id)$ algorithm to generate (pki, ski), returns pki to Adv and stores in an empty table T_{pk}.

2) $G_{sk}(pki)$:The oracle that generates the private key. Give as input pki by Adv. C searches pki from the table T_{pk} and returns the relevant to Adv.

3) $G_{rk}(pk_{DO}, pk_{Enc}, pk_{Dec})$: The oracle that generates the Rekey. Take three public keys as input, $pk_{DO}, pk_{Enc}, pk_{Dec}$ of the DO, the delegator EP and delegatee Dec. Firstly, C seeks the private key sk_{Enc} of the delegator EP from the T_{pk}. Then, C operates Rekey algorithm, returns Rekey to Adv.

4) $G_{reenc}(pk_{DO}, pk_{Enc}, pk_{Dec}, C_{owner})$: The oracle that Re-encryption. Take three public keys as input, $pk_{DO}, pk_{Enc}, pk_{Dec}$ of the DO, delegator EP and delegatee Dec respectively, an original ciphertext C_{owner} of the delegator EP, C first seeks the re-encryption key Rekey by running Rekey algorithm. Next, C operates Enc algorithm to compute C_{Enc} and sends C_{Enc} to Adv.

5) $G_{Dec}(C_{\cdot Enc}, pk_{Dec})$: If it is CPA, this oracle is refused to Adv. Now, the game $Exp_{\varepsilon,A}^{IND-DOMIBPRE-CPA}$ can be described as below:Game $Exp_{\varepsilon,A}^{IND-DOMIBPRE-CPA}$ (k):

   a. $Par \leftarrow Setup(k)$;

   b. $(pk^*_{DO}, pk^*_{DU}, m_0, m_1, sp) \leftarrow A_1^{G_{pk}, G_{sk}, G_{rk}, G_{reenc}, G_{Dec}}$, where$|m_0| = |m_1|$; (Find step)

   c. $d' \leftarrow_R \{0, 1\}$;

   d. $c^*_{Dec} = Enc(sk^*_{DO}, pk^*_{DU}, m_d)$;(Challenge step)

   e. $d' \leftarrow A_2^{G_{pk}, G_{sk}, G_{rk}, G_{reenc}, G_{Dec}}$, (par, c^*_{Dec}, SP); (Guess step);

   f. Return d';
   In the $Exp_{\varepsilon,A}^{IND-DOMIBPRE-CPA}$, the adversary Adv works in Find and Guess phase, respectively. Furthermore, some of limits are needed to Adv. Including:

      i. Adv is not permitted to make any private key generation queries either on pk^*_{DO}, pk^*_{DU}.

ii. *Adv* is not permitted to participate collusion attack. To be more precise, if *Adv* has made $G_{rk}(pk^*_{DO}, pk^*_{DU}, pk_{Dec})$ or $G_{reenc}(pk^*_{DO}, pk^*_{DU}, pk_{Dec}, c^*_{Dec})$ queries, then *Adv* is not permitted to make a $G_{sk}(pki)$ query, whereas the same, if *Adv* has made $G_{sk}(pki)$ query, *Adv* is not permitted to make $G_{rk}(pk^*_{DO}, pk^*_{DU}, pk_{Dec})$ or $G_{reenc}(pk^*_{DO}, pk^*_{DU}, pk_{Dec}, c^*_{Dec})$ queries.

iii. *Adv* is not permitted to initiate $G_{Dec}(c^*_{Dec}, pk^*_{Dec})$ query.

iv. *Adv* is not permitted to initiate $G_{Dec}(c, pk_{Enc})$ query, if $c = ReEnc(c^*_{Dec}, Rekey^*)$. Now, the advantage of *Adv* in the game $Exp^{IND-DOMIBPRE-CPA}_{\varepsilon,A}$ is defined:

$$\begin{aligned} & Adv^{IND-DOMIBPRE-CPA}_{\varepsilon,A}(k) \\ = \; & |2Pr[Exp^{IND-DOMIBPRE-CPA}_{\varepsilon,A}(k) \\ & = d] - 1| \end{aligned}$$

Definition 1. *We think DOMIBPRE scheme ε is IND-CPA secure if for all PPT algorithms A, we have $Adv^{IND-DOMIBPRE-CPA}_{\varepsilon,A}(k) \leq v(k)$, $v(.)$ is a negligible function.*

Notice 1. *The re-encrypted ciphertext is not offered to the adversary Adv, but we permit Adv to make re-encryption key generation query, therefore, Adv can convert the ciphertext by the re-encryption key. Nevertheless, the below situations are prohibited:*

1) If Adv has accessed to a Rekey from pk^*_{DU} to pk_{Enc}, then Adv is not permitted to make a private key query on pk_{Enc}.

2) If Adv has obtained private key of user pk_{Enc}, then Adv is not permitted to make Rekey query from pk^*_{DU} to pk_{Enc}.

5 System Description

In this section, we describe our concrete scheme design and smart contract design.

5.1 Scheme Description

1) $par \leftarrow setup(k)$: Input a security parameter k, Let e: $G_1 \times G_1 \rightarrow G_T$ is a bilinear map, g is a generator of G_T, let G_1 and G_T be two cyclic multiplicative groups of big prime order q. Selects two functions H_1 and H_2. $H_1\{0,1\}^* \rightarrow G_1$, $H_2 : G_T \rightarrow \{0,1\}^n$, $g_1 = e(g, g)$ selects $s \leftarrow z^*_q$, as the primary secret key (The generation of s is different from the traditional scheme, it is not generated by a single PKG, it is combined with multi PKG, each PKG contributes a portion of the private key, then the system aggregates them to generate the primary private key s, the detailed generation of is shown in Section 2.2 of [13]), output the system parameter $par(G_1, G_T, H_1, H_2, g_1, g, g^s)$.

2) $(pki, ski) \leftarrow keyGen(id)$: According to the ID of each user, MTA generates public key and private key for users.

$$\begin{aligned} sk_i &= H_1(id)^s \\ pk_i &= g^{ski} \end{aligned}$$

3) The encryption by Data Owner: This step consists of the following two sub-algorithms. Encrypt and TranskeyGen algorithms are run by DO.

a. $Encrypt.Owner \leftarrow (ski, pki, m)$:Data Owner chooses $t, \sigma \leftarrow z^*_q$ computes:

$$\begin{aligned} c_1 &= g^t \\ c_2 &= (m \oplus H_2(\sigma)) \cdot g_1^{1/sk_{DO}} \\ c_3 &= \sigma.e(g^s, H_1(id_{DO})) \\ c_4 &= pk_{Enc}^{1/sk_{DO}} \end{aligned}$$

Output: $C_{owner} = (c_1, c_2, c_3, c_4)$.
Data Owner uploads the C_{owner} to Encryption Proxy.

b. $TransKey \leftarrow TranskeyGen(id_{DO}, id_{DU})$:
Data Owner chooses$k \leftarrow z^*_q$, computes: TransKey:

$$< R_1, R_2 > = < C_1^{k/t}, H_1(id_{DO})^{-t} \cdot H_1(id_{DU})^k >$$

Data Owner returns the $TransKey$ to Data User.

4) The computation by Encryption Proxy: This step consists of two algorithms, RekeyGen and re-encrypt algorithms. They are all run by EP.

a. $Rekey \leftarrow RekeyGen(pk_{DO}, pk_{Enc}, pk_{Dec})$ computes:

$$\begin{aligned} Rekey &= (tk1_{DO \rightarrow Enc \rightarrow Dec}, tk2_{DO \rightarrow Enc \rightarrow Dec}) \\ &= ((pk_{DO}^{1/sk_{Enc}}, (pk_{Dec})^{1/sk_{Enc}})) \\ &= (g^{sk_{DO}/sk_{Enc}}, g^{sk_{Dec}/sk_{Enc}}) \end{aligned}$$

b. $Re - Encrypt.Enc \leftarrow C_{owner}$: Encryption Proxy computes:

$$\begin{aligned} c'_i &= c_i \qquad (i = 1, 2, 3) \\ c'_4 &= e(c_4, tk2_{DO \rightarrow Enc \rightarrow Dec}) \\ &= (g^{sk_{DO}/sk_{Enc}}, g^{sk_{Dec}/sk_{Enc}}) \end{aligned}$$

Output: $C_{Enc} = (c'_1, c'_2, c'_3, c'_4)$;
Encryption Proxy uploads the C_{Enc} to IPFS.

5) The decryption by Decryption Proxy: In this part, it consists of the following two sub-algorithms, ReEncrypt and Decrypt algorithms, both algorithms are

run on the Dec. The purpose of the ReEncrypt process is to convert the ciphertext C_{Enc} into a ciphertext C_{Dec} that DU can decrypt it by his or her own secret key. In order to reduce the computation of DU, decrypt algorithm is used for partial decryption of ciphertext. This step decrypts ciphertext C_{Dec} into ciphertext C'_{Dec}. Then, Dec sends the ciphertext C'_{Dec} to DU. The specific encryption processes are shown below:

a. $Decrypt.Dec \leftarrow (C_{Enc}, Transkey)$

$$\begin{aligned} C''_1 &= R_1 = g^k \\ C''_2 &= C'_2 \\ C''_3 &= C'_3.e(g^s, R_2) \\ C''_4 &= C'_4. \end{aligned}$$

Output: $C_{Dec} = (C''_1, C''_2, C''_3, C''_4)$.

b. $Decrypt.Dec \leftarrow C._{Enc}$ Decryption Proxy computes:

$$C_T = \frac{C''_2}{e(g, C_4^{''1/sk_{Dec}})}$$

6) $Decrypt.user \leftarrow C'_{Dec}$,Data User computes:

$$\begin{aligned} \sigma &= \frac{C''_3}{e(C''_1, sk_{DU})} \\ m &= C_T \oplus H_2(\sigma). \end{aligned}$$

Consistency. For the ciphertext $C'_{Dec} = (C''_1, C''_3, C''_T)$, where $C''_1 = g^k$;

$$\begin{aligned} C''_3 &= C'_3.e(g^s, R_2) \\ &= \sigma.e(g^s, H_1(id_{DO})^t).e(g^s, H_1(id_{DO})^{-t} \\ &\qquad \cdot H_1(id_{DU})^k) \\ &= \sigma.e(g^s, H_1(id_{DU})^k) \\ C''_4 &= e(C_4, pk_{Dec}^{1/sk_{Enc}}) \\ &= g^{sk_{Dec}/sk_{DO}} \\ C_T &= \frac{C''_2}{e(g, C_4^{''1/sk_{Dec}})} \\ &= \frac{(m \oplus H_2(\sigma)).g_1^{1/sk_{DO}}}{e(g, g^{1/sk_{DO}})} \\ &= m \oplus H_2(\sigma) \\ \sigma &= \frac{C''_3}{e(C''_1, sk_{DU})} \\ &= \frac{\sigma.e(g^s, H_1(id_{DU})^k}{e(g^k, H_1(id_{DU})^s)} \\ m &= C_T \oplus H_2(\sigma) \end{aligned}$$

5.2 Smart Contract Description

In this section, we present the smart contract algorithm logic employed in our paper. In the hyperledger fabric, smart contract is coded by Go language. There are three algorithms in our design. They are AddIndex, AddTxid and QueryIndex. We will introduce each of the four algorithms in the following:

1) AddIndex(Loc): This function can merely be performed by Data Owner(DO), when DO uploads some files to IPFS, DO can obtain Index Hash from IPFS, we call it Loc. According to smart contract we store the Loc in blockchain and record this transaction number txid.

Algorithm 1 AddIndex

```
Begin
Input: New Loc
Output:txid
if sender is not DO, then
    throw;
end if
New Loc has existed,then
return false;
Existing Index[New loc]⇐true;
return ture;
End
```

2) AddTxid(txid,IDO): This function can only be executed by DO. DO stores the txid in blockchain, and writes their ID as an index into the smart contract, by matching the ID, DO can obtain txid.

Algorithm 2 AddTxid

```
Begin
Input: txid,IDO
Output:bool
if sender is not DO, then
    throw;
end if
Mapping txid to IDO, and add it to extant Txid collection
return ture;
End
```

Algorithm 3 QueryTxid

```
Begin
Input: txid
Output:query Result
if sender is not authorized DU then then
    throw;
end if
Query the transaction content that corresponding to txid
Query Result⇐ Index[Loc]
return Query Result ;
End
```

3) QueryIndex(txid,IDU): DU submits a file sharing request to DO, after passing the identity authentication, DO returns txid to DU, DU uses txid as a condition to retrieve the corresponding data Loc in the blockchain network. According to the Loc, DU downloads the corresponding encrypted file in IPFS.

6 Security Proof

Theorem 1. *We assumed that mDBDH problem is difficult, then, our DOMIBPRE scheme proposed in 5 is IND-CPA security in the standard model.*
Proof: We require to attest that if there is a PPT adversary Adv has an advantage ω to attack DOMIBPRE scheme, and we need construct the other PPT adversary B, with the help of Adv, B can solve the mDBDH problem in G_1, with an advantage. Given a tuple $(g, g^a, g^b, g^c, T) \in G^4 \times G_T$, where $a, b \in z_q^*$ B can output 1 if $T = e(g,g)^{b/a}$ and 0 others.

The interaction between the two adversaries and B is presented as below:

Setup: B inputs a security parameter k to generate the system parameters $Par = (q, g, G, G_T, e, g_1)$ and return par to *Adv*, $g_1 = e(g, g)$.

Find: In this step, B answers the queries from *Adv* as below:

1) On a $G_{pk}(id)$ query: B chooses a random value $x_i \in z_q^*$, At the same time, B throws a weighted coin,$b_i \in$\{0,1\}, satisfying $b_i = 1$ with probability γ and 0 otherwise. If $b_i = 1$, B sets $pki = g^{ski}$, it signifies that the private key of user is ski. If $b_i = 0$, B sets $pki = (g^a)^{ski}$, it signifies that the private key of user i is αski. In this situation, does not know the private key either. Finally, B returns pki to *Adv* and stores (pki, ski, bi) in T_{pk}, where T_{pk} is a table that records public keys in the game and is vacant when initialized.
   For the sake of generality, assumed that *Adv* has made the suitable G_{pk} query before executing the following query:
2) On a $G_{sk}(pki)$ query : B searches the table T_{pk}, if $b_i = 1$, B returns ski to *Adv*, otherwise, a random number in z_q^* is output by B. B terminates.
3) On $G_{rk}(pk_{DO}, pk_{Enc}, pk_{Dec})$ query:
   Firstly, B searches T_{pk} to get $(pk_{DO}, sk_{DO}, b_{DO})$, $(pk_{Enc}, sk_{Enc}, b_{Enc}), (pk_{Dec}, sk_{Dec}, b_{Dec})$ and responds to *Adv* based on the below situation respectively.
   a. If $b_{Enc} = 1$, B calculates re-encryption key $Rekey = ((pk_{DO})^{1/sk_{Enc}}, (pk_{Dec})^{1/sk_{Enc}})$. If $(b_{DO}, b_{Enc}, b_{Dec}) = (0,0,0)$, B calculates re-encryption key $(g^{sk_{DO}/sk_{Enc}}, g^{sk_{Dec}/sk_{Enc}})$.
   b. If $b_{Enc} = 0$, and $(b_{DO}, b_{Dec}) \neq \{0,0\}$, B terminates.
4) On a $G_{reenc}(pk_{DO}, pk_{Enc}, pk_{Dec}, c_{owner})$ query: B makes a query to get Rekey, and runs the algorithm $ReEnc(C_{owner}, Rekey)$. According to the results return from the Rekey, B returns or terminates.

Challenge: *Adv* puts in two messages, $m_0, m_1 \in G_T$. *Adv* targets a sender pk_{DO}^*, and a target receiver pk_{DU}^*, with the below limits:

1) *Adv* does not make a query about the private key generation either on pk_{DO}^*, pk_{DU}^*, in Find step.
2) *Adv* does not initiate any $G_{rk}(pk_{DO}^*, pk_{DU}^*, pk_{Dec})$ query, where the private key of *Dec* is obtained to *Adv* by making an $G_{sk}(pk_{Dec})$query. B randomly chooses a bit $d \in \{0,1\}$, and retrieves T_{pk} to get $(pk_{DO}^*, sk_{DO}^*, b_{DO}^*)$. Then B calculates the ciphertext:

$$\begin{aligned} c_{Dec}^* &= (c_1^*, c_2^*, c_3^*, c_4^*) \\ &= (g^b, (m_d \oplus p).T^{1/sk_{DO}^*}, \\ &\quad \sigma.e(g^{s*}, H_1(id)_{DO}), (c_1^*)^{sk_{DU}^*/sk_{DO}^*}). \end{aligned}$$

   This process is viewed as DO to DU.
   Finally, c_{Dec}^* is transmitted to *Adv* as a challenge ciphertext.

Guess: More queries can be made by in the find phase, with the below limits:

1) *Adv* is not permitted to make any private key generation queries either on pk_{DO}^*, pk_{DU}^*.
2) If *Adv* has made $G_{rk}(pk_{DO}^*, pk_{DU}^*, pk_{Dec}^*)$ or $G_{reenc}(pk_{DO}^*, pk_{DU}^*, pk_{Dec}, c_{Dec}^*)$ queries, then *Adv* is not allowed to make a $G_{sk}(pki)$ query, whereas the same, if *Adv* has made a $G_{sk}(pki)$ query, *Adv* is not permitted to make $G_{rk}(pk_{DO}^*, pk_{DU}^*, pk_{Dec})$ or $G_{reenc}(pk_{DO}^*, pk_{DU}^*, pk_{Dec}, c_{Dec}^*)$ queries.

Decision: At this step, *Adv* outputs his guess $d' \in \{0,1\}$, if $d = d'$, B outputs 1, which manifests that $T = e(g,g)^{b/a}$, otherwise, B outputs 0, which manifesting that T is a random element in G_T.

Probability analysis that B does not abort:

1) On $G_{sk}(pki)$ query, B will not terminate in situation of bi = 1. Supposed *Adv* made qsk private key generation queries during the interaction, then the probability that B does not terminate in this situation is γ^{qsk}.
2) On $G_{rk}(pk_{DO}, pk_{Enc}, pk_{Dec})$query, B will not terminate in situation of *bi =1 or bi =bs=bj=0.* Supposed *Adv* made qrk re-encryption key queries, then the probability that B does not terminate is more γ^{qrk}.
3) On $G_{reenc}(pk_{DO}, pk_{Enc}, pk_{Dec}, C_{owner})$query, B will not terminate if there is a re-encryption key returned for this query. The probability analysis is as identical as the situation in $G_{rk}(pk_{DO}, pk_{Enc}, pk_{Dec})$query. Supposed that *Adv* can make qre re-encryption queries, then the probability that B does not terminate is more than γ^{qre}.

4) When the guess bit d_0 is output by Adv. B will not terminate if $b^*_{DU} = 0$ The probability in this situation is $1-\gamma$. It is distinct that if B does not terminate, the view of Adv during the interaction is as identical as the one in the actual attack. The total probability that B does not terminate is $f(\gamma) = (1-\gamma)\gamma^{qsk+qrk+qre}$. It is effortless to show that the function $f(\gamma)$ has a maximum value:

$$\frac{1}{qsk+qrk+qre} \times (1-\frac{1}{qsk+qrk+qre+1})^{qsk+qrk+qre+1}$$

at the $\gamma_{opt} = 1-1/(qsk+qrk+qre)$ using γ_{opt},the probability that B does not terminate is at least $\frac{1}{e(qsk+qrk+qre+1)}$ for large value of $qsk+qrk+qre$. This demonstrates that B has an advantage of at least $\varepsilon/e(qsk+qrk+qre+1)$.

7 Performance Analysis

In this section, some crucial features will be discussed. In order to assess the costs of this scheme in algorithm, we simulated our scheme with Pairing-Based Cryptography (PBC) library in C language (pbc-0.5.14). The elliptic curve parameter is chosen in Type-A, and the order of group is 160 bits. Using a computer with 64-bit windows 10 operation system, 2.60GHz Intel Core 8850H CPU, with 16 GB RAM. The computation cost of primitive cryptography operations shown in Table 2, E is represented the exponentiation operation in G or G_T, P is represented the pairing operation in G_T.

Table 2: Time cost of cryptography operations

Operation Names	T_E	T_p
$Times$	6.648	9.461

The performance comparison between our scheme and other related schemes is carried out results are shown in Table 3, from Table 3 we can see that [2, 28] and [32] did not use outsourcing computing, which brings a lot of computing overhead to users, in addition, these schemes did not realize the significance of multi authority. The scheme [31] deployed outsourced decryption and multi authority, but it lacks of completeness and flexibility. By comparing with the above schemes, it can be concluded that our scheme satisfies all properties.

In addition, we discussed the computation cost, we had a comparative summary of the computation costs in Table 4. From Table 4 we can see that our scheme reduced the computation of exponential and bilinear pairings, it is more efficient than other schemes (including ReKeyGen, Enc, ReEnc, and Dec). To make the comparison more intuitive, the number of computation is tuned from 0 to 100 in steps of 10 to record the running time of different schemes, as shown in Figure 7. We can see that in any calculation number, the running time of our scheme is improved compared to others. We also plotted the comparison of operation times in the bar chart, as shown in Figure 8, it can be seen that compared with [2,11,12,32], the operation times of our scheme has been reduced.

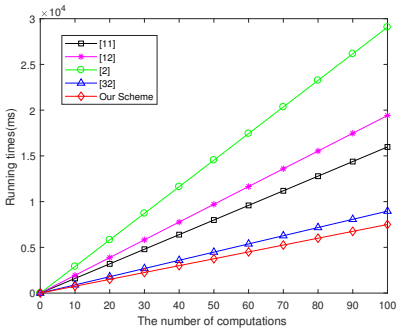

Figure 7: Comparison of time cost

Finally, by experimenting and comparing with other schemes, we illustrated that our scheme is more suitable for big data storage from the following three aspects:

1) **Storage cost:** According to reference [28], we get the cost of storing data on Ethereum, as show in Table 6: Obviously, as the amount of data stored increases, there is a huge storage cost, which is not suitable for big data storage. IPFS is a free storage platform for users, we also set up an IPFS network and test the time cost of IPFS, as shown in Table 5, its time cost can be accepted in practical. Therefore, IPFS is more suitable for storage big data.

2) **Trading efficiency:** Ethereum is a private blockchain. It is a completely decentralized organization and open organization, the consensus mechanism consumes a lot of energy, which affects its trading efficiency. Hyperledger fabric is a consortium blockchain, it is jointly managed by several institutions. We chose the hyperledger fabric as our blockchain network, we compared and analyzed the properties of Hyperledger Fabric and Ethereum, as shown in Table 7, hyperledger fabric has superior performance in TPS. Therefore, our scheme can enhance the efficiency of big data sharing.

3) **Data security:** Although IPFS provides the function of data encryption, the data owner was not involved in the whole process of data encryption, the data security can not be controlled by the data owner, in our scheme, before IPFS encrypted data, the data owner encrypts data by means of outsourcing, then they upload the encrypted data to IPFS system. Therefore, our scheme enhance the security of big data storage.

Table 3: Comparison of features

Schemes	Types of authority	Outsourced Encryption	Outsourced Decryption
[30]	Multiple	NO	YES
[23]	Single	NO	NO
[18]	Single	NO	NO
[4]	Single	YES	YES
Ours	Multiple	YES	YES

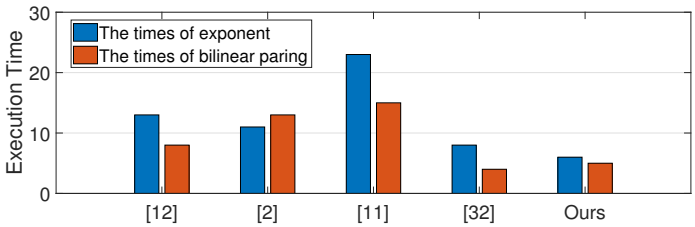

Figure 8: Comparison of computation time

Table 4: Comparison of computation cost

Schemes	[12]	[2]	[11]	[32]	Ours
ReKeyGen	1E	6E+1P	9E+1P	2E	2E
Enc	6E+1P	5E+1P	5E+1P	3E	3E+1P
ReEnc	4E+2P	9P	6E+7P	3P+E	1E+2P
Dec	2E+5P	2P	3E+6P	2E+1P	2P
Runtimes(ms)	159.772	194.141	290.679	89.588	86.133

Table 5: Time cost in IPFS

Names	Task Posting (Uploading)		Task Posting (Downloading)	
	Size(kb)	Time(ms)	Size(kb)	Time(ms)
Task_1	1158.39	495.41	1158.39	4.3265
Task_2	2357.45	607.56	2357.45	9.6875
Task_3	3562.81	736.82	3562.81	15.1238
Task_4	4823.76	912.49	4823.76	21.2926

8 Implementation of Blockchain Architecture

We simulated our blockchain scheme in personal computer, CPU 2.60 GHz, RAM 16GB. The operating system is ubuntu 16.04. We chose the hyperledger official test network fabric1.4.1 as our development platform and the solo as our consensus algorithm. Meanwhile, we build the blockchain network that consists of two organizations, four nodes and two channels. To illustrate our fabric blockchain system more clearly, the architecture of our scheme is shown in Figure 9.

As shown above, the member of nodes in Org 1 consist of data user, all data users join the channel A to get

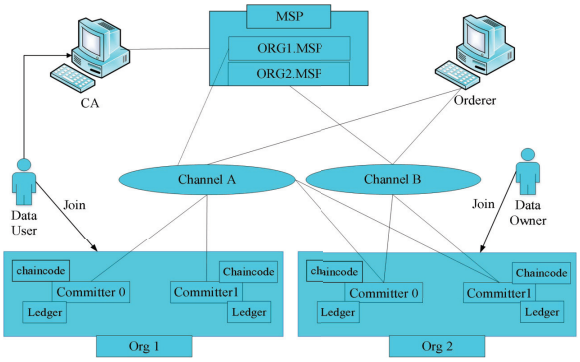

Figure 9: The architecture of our fabric network

Table 6: The cost of storing data

File Size(bytes)	Gas Used	Actual Cost(ether)	USD
208	28344	0.000056688	0.0236
1088	94984	0.000189968	0.0791
710	69280	0.00013856	0.0577

Table 7: The cost of storing data

Names	Type of blockchain	Consensus mechanism	participator	TPS
Ethereum	private chain	Pow/Pos/Dpos	Everyone	3-20
HyperledgerFabric	consortium chain	Solo/Kafka/Raft	Pre-Specified	1000-100000

the index of sharing files. At the same time, the member of nodes in Org 2 consists of data owners, they join the channel A and B simultaneously. Based on the smart contract, firstly, the index of the files which stored in IPFS were written in channel A by data owner, secondly, data owner records the transaction number tid_{An} which generated by the previous step to channel B. The data in channel B can only be accessed by data owner. When data user initiates a request to files, data owner returns the transaction number tid_{An} to data user after verifying his/her identity. According to tid_{An}, data user gets index of sharing files from the channel A.

Based on the hyperledger fabric blockchain network, we implement the aforesaid blockchain network. In order to make the operation of the client conveniently, we developed the software development kit (SDK) with go language. The interface and functions of the client are shown in the Figure 10.

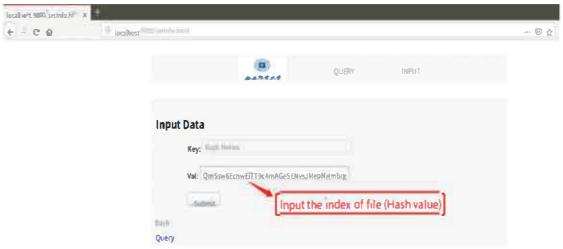

Figure 10: The process of uploading the index

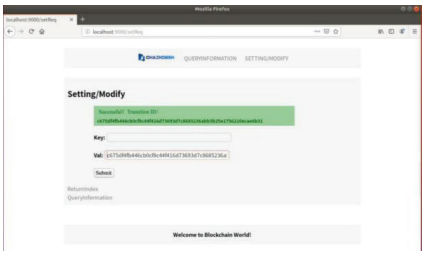

Figure 11: The feedback of storage in fabric

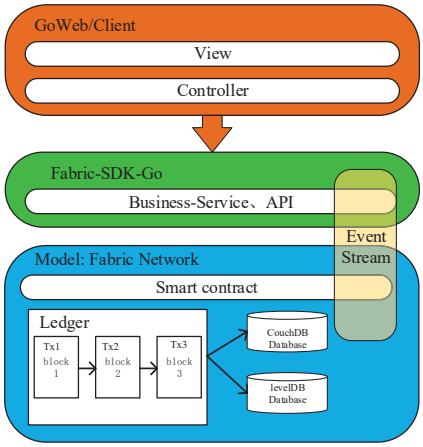

Figure 12: The architecture of MVC

As shown in Figure 10, data owner uploads the file index (hash string) into the blockchain system, when the data owner meets the conditions of smart contact, the system will return a transaction ID to client interface. As shown in Figure 11, when the data owner receives the transaction ID, the storage is successful. The experimental results prove our scheme is user-friendly. To explain clearly how smart contract help user upload the file index, we show the architecture of our MVC.

As shown in Figure 12, MVC architecture pattern consists of model, view and controller.

How to help DO upload the file index? When users upload file index from the view, they click the submission button, the controller will accept a requests from the client, it connects and invokes the corresponding API to access the smart contract through SDK. (In the step of uploading data, the smart contract which be invoked has the ability to write data in blockchain). When the requests meet the condition of smart contract, the system automatically executes the smart contract to write data in blockchain and sends the transaction ID to respond client. When the ID is returned, the storage is successful.

9 Conclusion

In this paper, a novel IBPRE scheme, DOM-IBPRE scheme is proposed and demonstrated, which is achieved by combining IBPRE, blockchain and the inter planetary file system (IPFS) technology. The scheme can avoid the complexity of certificate management, enhance the security of big data storage and ameliorate the efficiency of big data sharing. In addition, a single PKG is replaced by multi trusted authority in our scheme, the problem that the malicious attack on the PKG results in the leak of the private key of each user in conventional IBPRE scheme is solved. At the same time,we also assessed the security performance of the scheme by Chosen-Plaintext Attack (CPA) based on a modified DBDH problem in the standard model and simulated our scheme with Pairing-Based Cryptography (PBC) library, by comparing with other PRE schemes, our scheme has the better performance in computation. Finally, we implemented our blockchain architecture under linux system with the hyperledger official test network fabric, based on the smart contract, the index of sharing file is stored in blockchain, all authorized users in this system can store and read data in blockchain through the web. The test results validate that our scheme is practical.

Acknowledgments

This work was supported by the National Key R&D Program of China under Grant 2017YFB0802000, the Natural Science Foundation of China under Grant 61802303, 61772418 and 61602378, the Key Research and Development Program of Shaanxi under Grant 2019KW-053, the Innovation Ability Support Program in Shaanxi Province of China under Grant 2017KJXX-47, the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2019JQ-866, 2018JZ6001 and 2016JM6033, the Research Program of Education Bureau of Shaanxi Province under Grant 19JK0803, the New Star Team of Xi?an University of Posts and Telecommunications under Grant 2016-02, the Fundamental Research Funds for the Central Universities under Grant GK201903005, and Guangxi Cooperative Innovation Center of Cloud Computing and Big Data under Grant YD1903.

References

[1] A. Banerjee, "Blockchain technology: Supply chain insights from ERP," in *Advances in Computers*, vol. 111, pp. 69–98, 2018.

[2] J. Bankar and J. Raghatwan, "Identity based proxy re-encryption using forward security in cloud framework," in *International Conference on Computing, Communication, Control and Automation (ICCUBEA'17)*, pp. 1–5, 2017.

[3] J. Benet and N. Greco, "Filecoin: A decentralized storage network," *Protocol Labs*, pp. 1–36, 2018. (`https://research.protocol.ai/publications/filecoin-a-decentralized-storage-network/`)

[4] G. Chunpeng, Z. Liu, and J. Xia, "Revocable identity-based broadcast proxy re-encryption for data sharing in clouds," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, 2019.

[5] D. Hopwood, S. Bowe, T. Hornby, N. Wilcox, *Zcash Protocol Specification*, Ver. 2021.2.7, NU5 Proposal, June 28, 2021. (`https://raw.githubusercontent.com/zcash/zips/master/protocol/protocol.pdf`)

[6] E. Androulaki, B. Artem, and V. Bortnikov, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, pp. 1–15, 2018.

[7] W. G. Ethereum, "A secure decentralised generalised transaction ledger," *Bitcoin*, vol. 151, pp. 1–32, 2014. (`https://gavwood.com/paper.pdf`)

[8] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *International Conference on Applied Cryptography and Network Security*, pp. 288–306, 2007.

[9] H. Guo, Z. Zhang, and J. Xu, "Accountable proxy re-encryption for secure data sharing," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 1, pp. 145–159, 2018.

[10] J. Bankar and J. Raghatwan, "Identity based proxy re-encryption using forward security in cloud framework," in *International Conference on Computing, Communication, Control and Automation (ICCUBEA'17)*, pp. 1–5, 2017.

[11] K. Liang, W. Susilo, and J. K. Liu, "Efficient and fully CCA secure conditional proxy re-encryption from hierarchical identity-based encryption," *The Computer Journal*, vol. 58, no. 10, pp. 2778–2792, 2015.

[12] B. Libert and D. Vergnaud, "Unidirectional chosen-ciphertext secure proxy re-encryption," in *International Workshop on Public Key Cryptography*, pp. 360–379, 2008.

[13] Y. Cheng, W. Luo, S. Wen, "Blockchain-based electronic health record sharing scheme," *Journal of Computer Applications*, vol. 40, no. 1, pp. 157–161, 2020.

[14] B Lynn, *PBC Library - Pairing-Based Cryptography Library*, June 29, 2021. (`https://github.com/blynn/pbc`)

[15] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 127–144, 1998.

[16] L. Mand, N. Kannouf, and Y. Chahid, "Blockchain-based pki for content-centric networking," in *The Proceedings of the Third International Conference on Smart City Applications*, pp. 656–667, 2018.

[17] A. Manzoor, M. Liyanage, and A. Braeke, "Blockchain based proxy re-encryption scheme

for secure IoT data sharing," in *IEEE International Conference on Blockchain and Cryptocurrency (ICBC'19)*, pp. 99–103, 2019.

[18] Y. N. Li, X. Feng, and J. Xie, "A decentralized and secure blockchain platform for open fair data trading," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 7, p. e5578, 2020.

[19] S. Nakamoto, "A peer-to-peer electronic cash system," *Ethereum Project Yellow Paper*, 2008. (`https://bitcoin.org/bitcoin.pdf`)

[20] M. Pilkington, "Blockchain technology: Principles and applications," in *Research Handbook on Digital Transformations*, 2016. DOI:10.4337/9781784717766.00019.

[21] S. Ranjan, A. Negi, and H. Jain, "Network system design using hyperledger fabric: Permissioned blockchain framework," in *The Twelfth International Conference on Contemporary Computing (IC3'19)*, pp. 1–6, 2019.

[22] S. Rezaei, M. A. Doostari, and M. Bayat, "A lightweight and efficient data sharing scheme for cloud computing," *International Journal of Electronics and Information Engineering*, vol. 9, no. 2, pp. 115–131, 2018.

[23] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 457–473, 2005.

[24] J. Shao, G. Wei, and Y. Ling, "Identity-based conditional proxy re-encryption," in *IEEE International Conference on Communications (ICC'11)*, pp. 1–5, 2011.

[25] D. Vorick and L. Champine, *Sia: Simple Decentralized Storage*, 2014. (`https://sia.tech/sia.pdf`)

[26] Q. Wang, W. Li, and Z. Qin, "Proxy re-encryption in access control framework of information-centric networks," *IEEE Access*, vol. 7, pp. 48417–48429, 2019.

[27] S. Wang, X. Wang, and Y. Zhang, "A secure cloud storage framework with access control based on blockchain," *Ieee Access*, vol. 7, pp. 112713–112725, 2019.

[28] S. Wang, Y. Zhang, and Y. Zhang, "A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems," *IEEE Access*, vol. 6, pp. 38437–38450, 2018.

[29] S. Wilkinson, T. Boshevski, and J. Brandoff, *Storj a Peer-to-Peer Cloud Storage Network*, 2014. (`https://www.storj.io/storj2014.pdf`)

[30] P. Xu, T. Jiao, and Q. Wu, "Conditional identity-based broadcast proxy re-encryption and its application to cloud email," *IEEE Transactions on Computers*, vol. 65, no. 1, pp. 66–79, 2015.

[31] K. Yang and X. Jia, "Expressive, efficient, and revocable data access control for multi-authority cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1735–1744, 2013.

[32] P. Zeng and K. K. R. Choo, "A new kind of conditional proxy re-encryption for secure cloud storage," *IEEE Access*, vol. 6, pp. 70017–70024, 2018.

[33] Q. Zheng, Y. Li, and P. Chen, "An innovative ipfs-based storage model for blockchain," in *IEEE/WIC/ACM International Conference on Web Intelligence (WI'18)*, pp. 704–708, 2018.

[34] G. Zyskind and O. Nathan, "Decentralizing privacy: Using blockchain to protect personal data," in *IEEE Security and Privacy Workshops*, pp. 180–184, 2015.

Biography

Jiayu He received the B.S. degree from the Xi'an University of Posts and Telecommunications in 2018. He is currently pursuing the M.S. degree with the National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, China. His research interests include blockchain technology and security in cloud storage

Dong Zheng received the Ph.D. degree from Xidian University in 1999. He joined the School of Information Security Engineering, Shanghai JiaoTong University. He is currently a Professor with the Xi'an University of Posts and Telecommunications, China. His research interests include information theory, cryptography, and information security. He is a Senior Member of the Chinese Association for Cryptologic Research and a member of the Chinese Communication Society.

Rui Guo received the Ph.D. degree from the State Key Laboratory of Networking and Switch Technology, Beijing University of Posts and Telecommunications, China, in 2014. He is currently a Lecturer with the National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications. His present research interests include attribute-based cryptograph, cloud computing, and blockchain technology.

Yushuang Chen received the B.S. degree from the Xi'an University of Posts and Telecommunications in 2018. She is currently pursuing the M.S. degree with the National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, China. Her research interests include blockchain, information security, and the Internet of Things (IoT).

Kemeng Li received the bachelors degree from Xi'an University of Posts and Telecommunications in 2018. He is currently pursuing his master degree at National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications. His research interests include blockchain technology and security in cloud storage.

Xiaoling Tao received the M.S. degree in computer application technology from Guilin University of Electronic Technology. She is a professor at the school of computer science and information security, Guilin University of Electronic Technology. Her research interests include cloud computing and security and network security.