Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 21(6): 6167–6189.
DOI: 10.3934/mbe.2024271
Received: 21 November 2023
Revised: 10 March 20242
Accepted: 16 April 2024
Published: 03 June 2024

Research article

An anti-impersonation attack electronic health record sharing scheme based on proxy re-encryption and blockchain

Jiayuan Zhang[1], Rongxin Guo[2,*], Yifan Shi[2] and Wanting Tang[1]

[1] College of Information Science and Engineering, Huaqiao University, Xiamen 361021, China
[2] College of Engineering, Huaqiao University, Quanzhou 362021, China

* **Correspondence:** Email: grxeee@hqu.edu.cn; Tel: +8613959898683.

Abstract: Many current electronic medical record (EMR) sharing schemes that use proxy re-encryption and blockchain do not fully consider the potential threat of malicious node impersonation attacks. This oversight could lead to data leakage as attackers masquerade as legitimate users or proxy nodes during the sharing process. To deal with this problem, we propose an EMR sharing scheme based on proxy re-encryption and blockchain to protect against impersonation attacks. First, we prevent the potential threat of impersonation attacks by generating a shared temporary key and assigning tasks to multiple proxy nodes. Second, we use a random function to ensure that the selection of encrypted proxy nodes is fair. Third, we use a combination of blockchain and the InterPlanetary File System to solve the problem of insufficient storage capacity of shared processes and ensure the storage security of EMRs. Through the security proof, our scheme guarantees anti-impersonation, anti-collusion, and anti-chosen plaintext attack capability in the sharing process of EMRs. Additionally, experiments on the blockchain platform, namely Chain33, show that our scheme significantly increases efficiency.

Keywords: anti-impersonation attack; blockchain; electronic medical record; proxy re-encryption; data security

1. Introduction

In the current highly information-based society, data sharing has the advantages of promoting social development and improving the efficiency of information circulation. However, there are many potential risks in the actual process of data sharing, and impersonation attacks constitute one of the main attack methods. Illegal attackers impersonate legitimate users to obtain unauthorized access rights, and then access shared data or perform malicious operations [1–3]. If no effective means can be implemented to solve the problem, the attacker may steal sensitive information, violate the privacy of data users, and even tamper with shared data, resulting in the data receiver performing wrong

decisions due to incorrect data. Electronic medical records (EMRs), as one type of shared data, are widely used in the medical field. If the masquerading attack cannot be restricted, it will endanger the privacy security of patients, and it is not conducive to obtain a correct diagnosis for medical services [4]. It may also cause damage to the credibility of medical institutions, even leading to legal problems in serious cases.

In the process of EMR sharing, in order to prevent the impersonation attacks from illegal users, strong authentication of multiple factors is often used to ensure the credibility of the identity of the sharing parties. Although accurate access verification can restrain impostors to a certain extent, there are still some cases in which attackers can successfully pass the verification test. Therefore, it is often necessary to encrypt the EMR during the sharing process to ensure that even if the attacker has access to the EMR, they cannot easily read or tamper with its content. However, the encryption of EMRs also needs to address the risk of encryption key leakage. If simple symmetric encryption is used to solve it, once the symmetric key is illegally eavesdropped, the problem of data security still exists [5]. Although the asymmetric key can alleviate it to a certain extent, its nature will cause the sharing process to become cumbersome. Therefore, how to effectively ensure the security of EMR sharing is worthy of our in-depth study.

As a new technology, blockchain has been applied in many scenarios such as finance, the Internet of Things (IoT), credit information and ownership management [6]. It has the characteristics of decentralization, traceability and immutability. In the medical record sharing scenario, it can also enable and engender shared EMR security, trustworthiness and source traceability. Transparency and security for medical applications can be achieved through blockchain, contributing to a reliable and secure healthcare system [7]. As an extension of cryptography, proxy re-encryption technology is mainly used to solve the problems of data security and privacy protection in the process of ciphertext sharing. In proxy re-encryption, the ciphertext encrypted by the data owner can be converted into another ciphertext by the Re-encryption key (ReKey), and the converted ciphertext can be decrypted by the private key of the data requester. At the same time, the proxy cannot obtain any information about the corresponding plaintext throughout the whole process of ciphertext conversion.

In view of the above problems, we proposes an EMR sharing scheme based on proxy re-encryption and blockchain. First, we added a shared temporary key in the shared encryption process, and divided the labor of multiple proxy nodes to prevent the potential harm of impersonation attacks and resist collusion attacks. Second, we have guaranteed the fairness of the selection of encrypted proxy nodes through the use of a random function to ensure that the task allocation to multiple proxy nodes is randomly fair. Third, we have applied a combination of blockchain and the InterPlanetary File System (IPFS) to solve the problem of insufficient storage capacity of shared processes and ensure the storage security of EMRs. Through the security proof, our scheme guarantees anti-impersonation, anti-collusion, and anti-chosen plaintext attack capability in the sharing process of EMRs. Additionally, experiments on the blockchain platform, namely Chain33, show that our scheme considerably improves efficiency.

The rest of this paper is structured as follows. The Section 2 introduces the literature review. Section 3 puts forward the scheme design, including the scheme model, scheme design for anti-impersonation attack, the system algorithm and an application scenario. Section 4 proves the security of this scheme. Section 5 describes the experimental results and discussion in detail. Finally, in Section 6, the full text is summarized and the future research direction is discussed.

2. Literature review

2.1. Blockchain technology

In recent years, the development of blockchain technology has provided new ideas for solving the secure storage and management of medical data. The storage and circulation of EMRs are often limited within hospitals, and the "information island" of data can significantly inconvenience patients when visiting different hospitals. The birth of blockchain technology can overcome this limitation. An EMR storage scheme based on blockchain can store data in the decentralized blockchain network, thereby avoiding the security risks in the centralized database storage method. At the same time, blockchain technology can also ensure the non-tampering and traceability of data, thereby enhancing the security and credibility of medical data [8]. Different from the traditional electronic medical platform, there is no centralized organization in the blockchain system, and all nodes jointly maintain the security of the system. The consensus mechanism coordinates all nodes to reach the only result accepted by all nodes, so as to realize the data consistency of each node and ensure the authenticity of the blockchain data. Therefore, the use of blockchain can solve the problems of trust and security in the sharing process [9]. Neela and Kavitha [10] proposed a blockchain-driven encryption scheme that employs chaotic deep generative adversarial networks to bolster the limited image encryption capabilities of conventional cloud storage systems. The primary objective was to fortify the secure storage of medical images and safeguard them from potential hacker attacks. Qu [11] proposed a robust and practical Byzantine fault-tolerant system for managing medical information on the blockchain. The primary objective of this system is to ensure the integrity and confidentiality of medical data, guarding against unauthorized tampering and leakage. Huang et al. [12] proposed a privacy-preserving framework built upon blockchain technology. This approach utilizes zero-knowledge proofs to validate if a patient's medical data satisfy certain predefined criteria set forth by a research institution. By adopting this method, secure data sharing can be accomplished among multiple entities without compromising patient privacy. Liu et al. [13] proposed an EMR storage and sharing scheme, where the nodes in the scheme possess both traceability and anonymity. Furthermore, the scheme utilizes a decentralized consortium blockchain to address the issue of malicious nodes being unidentifiable in a completely anonymous sharing schemes. Wang et al. [14] proposed a blockchain-based EMRs sharing scheme by utilizing attribute-based encryption (ABE) with fixed-size attributes. This scheme incorporates the access policy directly into the search results of the blockchain, enabling authorized users to conduct multi-keyword Boolean searches on encrypted EMRs. Marichamy and Natarajan [15] proposed a cryptographic hash generator technology for secure and trusted data storage and transmission based on blockchain in the Hadoop Distributed File System. Medical data are divided into sensitive data and insensitive data, where sensitive data are encrypted by discrete shear wave transformation, and stored in the blockchain to enhance the security level.

2.2. Anti-impersonation attack technology

Although the emergence of blockchain technology has provided many conveniences for data sharing, its openness and the use of asymmetric keys also make data sharing susceptible to impersonation attacks. In order to address the security vulnerabilities caused by impersonation attacks, Yang et al. [16] proposed an enhanced blockchain-based non-pairing certificateless signature

scheme. This scheme introduces a new random value ω_i into the signature generation algorithm and utilizes the hash function value μ_i to bind system public keys and user public keys and consequently mitigate the risks of general impersonation attacks. To defend against public key replacement attacks and joint attacks by malicious sensor nodes, Yang et al. [17] optimized the unpaired certificateless aggregate signature scheme. The enhanced scheme adopts fixed-length aggregate signatures, effectively reducing transmission bandwidth. Kholidy [18] proposed a method to detect cloud computing impersonation attacks by associating user behaviors in different environments, analyzing sequences of relevant system calls from virtual machine operating systems, and using NetFlow data from the network environment. Neural networks are used for integrated inference to produce better detection results. Yang et al. [19] proposed a heterogeneous signature encryption scheme from public key infrastructure to identity-based cryptography, and they conducted a multi-ciphertext equivalence test in a Network of vehicles to ensure that inter-vehicle communication data are protected from illegal access and impersonation attack by malicious vehicles. Ma et al. [20] designed a novel cloud-based industrial IoT deployment scheme with an unpaired dual-server setup, eliminating the need for bilinear pairs and the use of secure channels to provide users with security guarantees against impersonation attacks. Zhang and Zhou [21] developed an IND-CCA secure multi-authority ciphertext policy ABE scheme with outsourced decryption and a progressive mode by using zero-knowledge proofs to protect users' secrets from being leaked to servers. Due to the randomness of authentication messages, it can resist impersonation attacks by malicious servers.

2.3. Proxy re-encryption technology

In traditional approaches to EMRs storage, symmetric encryption or double encryption is commonly employed. However, the encrypted data can only be decrypted by the data owner, impeding effective data sharing. Therefore, an innovative encryption scheme is imperative to enable secure data sharing. In 1998, Blaze et al. [22] proposed the concept of proxy re-encryption at the European Cryptography Conference. This approach allows encrypted data to be shared without revealing the encryption keys. The process involves transforming the ciphertext through the use of a semi-trusted proxy node. Prior to performing this transformation, a ReKey needs to be generated and sent to the proxy node. Once the ReKey is received, the proxy node can re-encrypt the ciphertext and forward it to the intended recipient. Decryption of the re-encrypted ciphertext into plaintext is only possible by using the recipient's private key, thus achieving the objective of data sharing. Throughout the entire ciphertext transformation process, the proxy node remains unable to access any information about the corresponding plaintext. Guo et al. [23] proposed a non-interactive accountable proxy re-encryption scheme. This method introduces a mechanism to verify if the proxy is misusing the ReKey to access the data owner's decryption capability. Fan et al. [24] proposed an innovative scheme for cloud computing known as timed-release proxy conditional re-encryption. This scheme allows individuals to share specific files with others, but with the added flexibility of setting time constraints on the sharing process. Azbeg et al. [25] proposed an advanced medical system that combines the IoT with blockchain technology to ensure strong security. The integrated system uses proxy re-encryption to enhance security, and it uses proxy nodes to store hash data and thus guarantee the privacy and authenticity of sensitive EMRs. Manzoor et al. [26] proposed a cutting-edge market for sharing IoT data based on blockchain technology. This innovative system integrates smart contracts with proxy re-encryption technology, ensuring that data visibility is restricted solely to the data owners and

authorized personnel within the smart contract, thus providing a secure solution for the storage and management of sensor data.

In traditional proxy re-encryption, public key certificates are required to authenticate the identity of users, and the management and storage of certificates undoubtedly increase implementation difficulties. Therefore, in 2007, Green and Ateniese [27] proposed an identity-based proxy re-encryption (IBPRE) scheme. This scheme allows the proxy to efficiently transform ciphertext that has been encrypted by using user A's identity information into ciphertext encrypted by using user B's identity information directly. Then this encryption scheme was continuously extended, Wang et al. [28] proposed an IBPRE scheme that achieves unidirectionality, versatility, and CCA2 security through the use of random padding techniques. Xiong et al. [29] proposed a perforated identity-based scheme that incorporates a message server as a proxy for ciphertext transformation among group participants. This approach allows the receiver to selectively revoke their decryption capability for a specific message without impacting other messages within the group to semicolon, effectively safeguarding the security and privacy of group messages ensures that sensitive information remains protected and accessible only to authorized recipients. Maiti and Misra [30] proposed an innovative approach to IBPRE, enhancing the confidentiality of the receiving group's identities in the re-encrypted broadcast ciphertext and thereby further enhancing user privacy. Ge et al. [31] proposed a revocable IBPRE scheme to address the issue of data owners' inability to leverage cloud computing advantages in traditional IBPRE schemes for efficient shared key generation. The scheme allows the proxy server to revoke a specified list of delegations made by data owners who use re-encryption keys and generate new shared keys, thereby fully leveraging the efficiency of cloud computing for secure sharing with revocation. Zhou et al. [32] proposed an IBPRE scheme with cryptographic reverse firewall to address the problem of open networks and semi-trusted cloud service providers exposing users' private medical data to backdoor attackers. They used identity attributes to solve the problem of attackers conducting infiltration attacks through the data sharing process. Lin et al. [33] proposed an IBPRE scheme for fog computing scenarios. The scheme utilizes anonymous techniques to generate encryption keys and employs a public channel for key distribution, thus avoiding the cumbersome issue of key escrow. Xu et al. [34] proposed an IBPRE scheme, allowing the sender to encrypt messages by explicitly specifying the identities of multiple recipients. This approach empowers the sender to delegate the ReKey to a proxy node, facilitating the conversion of the initial ciphertext into a new ciphertext that is suitable for a distinct group of intended recipients. Yao et al. [35] proposed a revocable conditional IBPRE scheme with ciphertext evolution, which supports updating ciphertexts to adapt to new identity keys and achieving authorization revocation through ciphertext evolution after identity keys are changed. Kan et al. [36] proposed an IBPRE scheme for duplicate data elimination, it combines cloud duplicate data elimination with access control, allowing the cloud server to eliminate redundant data based on user identity and ownership to save storage space. This overcomes the difficulty that is typically associated with traditional data deduplication technology when eliminating duplicate encrypted data.

3. Methodology design

3.1. Scheme model

Figure 1 shows the proposed EMR sharing scheme model, which is mainly composed of five parts, namely, a storage server, data owner, data requester, smart contract and proxy.

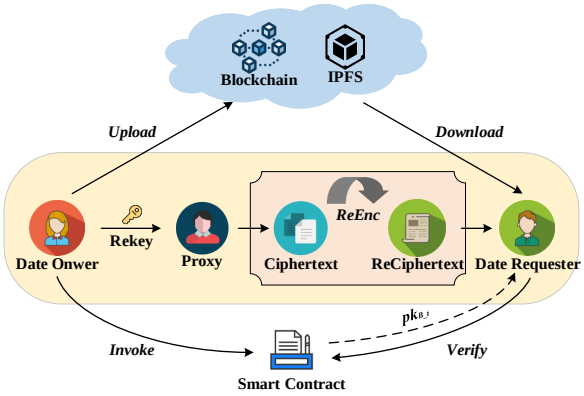

Figure 1. Scheme model of EMR sharing system.

1) Storage server: The storage server consists of blockchain and the IPFS. According to the application scenario for this scheme, medical institutions and regulatory authorities are used as consensus nodes to form the consortium blockchain. As a trusted platform, blockchain interacts with and records all entities in the system. However, due to the limited storage space of the consortium blockchain, the data storage cannot be completed alone, so the decentralized IPFS cluster is used to store the encrypted electronic prescription. The initial ciphertext C_{date} of the EMR is generated by offline symmetric encryption and stored in the IPFS. The IPFS hash index $H_{IPFS} = H_{256}(H_{256}(block_1) \parallel ... \parallel H_{256}(block_n))$ and the symmetric key of the file are used as plaintext, which is defined as $M = key \parallel H_{IPFS}$, and the plaintext is hashed and stored to the blockchain. This storage method has the advantages of non-repudiation and immutability.

2) Data owner: The patient has absolute control over the electronic health record. The data owner initiates the sharing of EMRs by calling the smart contract, sets the access conditions to realize the fine-grained division of access to encrypted data, and generates a shared public key set according to the public key of the data requesters.

3) Data requester: Usually doctors, scientific research institutions or medical institutions request data. When the data requester initiates a request for access to the shared medical record, they need to pass the access permission verification of the smart contract first, and then it can obtain the corresponding shared public key and the re-encrypted ciphertext sent by the proxy node, so as to decrypt the data sharing.

4) Smart contract: It is mainly used to access the verification contract and randomly select the contract. A smart contract is a computer protocol that provides verification and performs automatic execution of contracts, which can provide access restrictions and improve sharing efficiency. In order to prevent frequent access applications from malicious nodes from interfering with the normal sharing of users' EMRs, the smart contract is called to realize the access restriction and fine-grained division

of EMRs access.

5) Proxy: It is mainly divided into encryption proxy node and transmission proxy node. The encrypted proxy node is generated by randomly selecting contracts, which will be selected again every certain interval, and it is responsible for the proxy re-encryption of EMRs. The transmission proxy nodes are the remaining proxy nodes that are not selected and are responsible for transmitting the rest of the heavy ciphertext. Since the proxy node is a semi-trusted node, its behavior will be supervised by the regulatory department. Once the malicious behavior occurs, the proxy will be disqualified.

3.2. Anti-impersonation design for the scheme

In order to prevent potential impersonation attacks in the process of sharing data, we mainly designed a proxy re-encryption scheme with anti-impersonation capability to protect the sharing security of EMRs. In the process of proxy re-encryption, it generates a temporary shared key and randomly assigns tasks to multiple proxy nodes to prevent impersonation attacks.

The shared public key $pk_{\lrcorner t}$ is generated based on the current timestamp T_{now} and the user's public key; the shared private key $sk_{\lrcorner t}$ is generated through the use of a private key generator, and the shared key is only used for this medical record information sharing. Each proxy node $P_i(i = 1, 2, \ldots, n)$ needs to generate a random number $\eta_i(i = 1, 2, \ldots, n)$ every time interval and send it to the random selection contract; the contract completes the random selection of the encrypted proxy node through calculation. After receiving the random numbers sent by all proxy nodes, the contract scrambles and concatenates all the random numbers and calculates the hash value $H_\mu = \text{hash}\,(\eta_1 \parallel \eta_2 \parallel \cdots \parallel \eta_n)$ of all the random numbers μ; it then determines the proxy node candidate for proxy re-encryption by mapping the hash value H_μ to the address list of the proxy node; the rest of the selected proxy nodes are used as the transmission nodes of this encryption. By increasing the randomness of proxy node selection, it reduces the possibility of collusion in the sharing process and reduces the possibility of proxy nodes being attacked.

After receiving the public key $pk_{B\lrcorner t}$ shared by data requester Bob, the data owner generates the ReKey $Rekey_{A\to B} = (Rk_1 \parallel Rk_2)$ and splits it. Only part of the ReKey Rk_2 and ciphertext C are sent to the encryption proxy node for encryption, and then the partial ReKey Rk_1 is split and sent to the rest of the proxy nodes. After receiving the data, the encrypted proxy node is able to encrypt the ciphertext C and output the result to Bob, while the remaining proxy nodes send part of the ReKey Rk_1 fragment to Bob. After receiving C', Bob decrypts the plaintext M through the use of the shared private key $sk_{B\lrcorner t}$ that they own. Figure 2 shows the EMR sharing process:

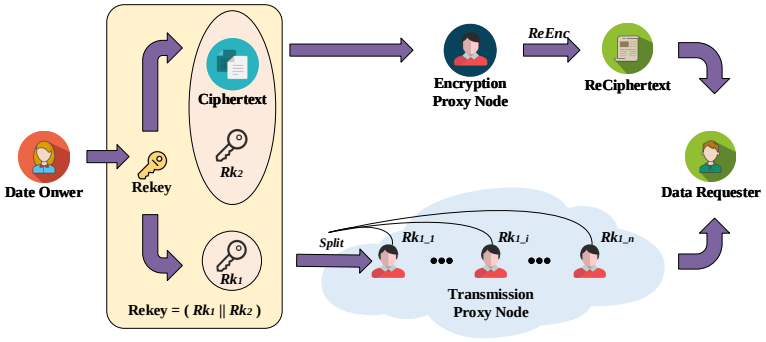

Figure 2. The EMR sharing process.

3.3. System algorithm

Table 1. Algorithm related symbolic description.

Notation	Description
ID	User identity information
M	Plaintext
$P_i(i = 1, 2, \ldots, n)$	Proxy nodes
(pk_A, sk_A)	The key pair of the data owner Alice
(pk_B, sk_B)	The key pair of the data requester Bob
(pk_{A_t}, sk_{A_t})	The shared key pair of the data owner Alice
(pk_{B_t}, sk_{B_t})	The shared key pair of the data owner Bob
T_{limit}	Shared EMR access timeliness
T_{now}	The current timestamp
C_{date}	Initial ciphertext
C	Encrypting ciphertext
$C' = C'_1 \parallel C'_2 \parallel C'_3$	Re-encrypt the ciphertext
$Rekey_{A\to B} = (Rk_1 \parallel Rk_2)$	Re-encryption key
key	Symmetric key
H_μ	Main random number hash
H_{IPFS}	Initial ciphertext hash
H_M	Plaintext hash
H'_M	Decrypted plaintext hash

1) $Setup(\lambda) \to (params, msk)$
The input includes a security parameter λ, an additive cyclic group G_1 and a multiplicative cyclic group G_T, both of which have the order of p. The bilinear mapping is denoted as $e : G_1 \times G_1 \to G_T$, and g is the generator of G_1. Two collision-resistant hash functions $H_1 : \{0,1\}^* \to G_1$ and $H_2 : G_T \to G_1$ are chosen. The output is the system's public parameter $params = (G_1, G_T, p, g, e, H_1, H_2)$, and a random number $s \in Z_p^*$ is selected as the master secret key $msk =$

$\{s\}$ for the key generator PKG. The system's public key is denoted as $mpk = \{g, g^s\}$.

2) $KeyGen(param, msk, ID) \rightarrow (pk, sk)$
Alice and Bob respectively use their identity information such as user their ID to generate public and private key pairs (pk_A, sk_A) and (pk_B, sk_B). The calculation is as follows:

$$pk = H_1(ID) \tag{3.1}$$

$$sk = H_1(ID)^s \tag{3.2}$$

3) $ShareKeyGen(params, msk, pk, sk) \rightarrow (pk_{\_t}, sk_{\_t})$
After Alice, i.e., the owner of EMR, completes the release of the smart contract, the contract is mapped to Z_p^* based on the hash calculation of the current timestamp $T_{now} \in \{0,1\}^*$, and the mapping result is recorded as $t \in Z_p^*$. The process to generate Alice's shared public key pk_{A_t}, and the shared public key of users allowed access, such as Bob's pk_{B_t}, is computed as follows:

$$pk_{\_t} = H_1(ID) \cdot g^t \tag{3.3}$$

The shared private key is generated by the PKG, which is calculated as follows:

$$sk_{\_t} = (pk_{\_t})^s \tag{3.4}$$

4) $KeyEncrypt(params, pk_{\_t}, M) \rightarrow C$
Alice chooses a random number $r \in Z_P^*$, and encrypts M by using pk_{A_t}. The encryption calculation is as follows:

$$C_1 = g^r \tag{3.5}$$

$$C_2 = M \cdot e(g^s, pk_{A\_t})^r \tag{3.6}$$

$$C = C_1 \parallel C_2 \tag{3.7}$$

The ciphertext C can only be decrypted and obtain M through the use of Alice's sk_{A_t}. The calculation process is as follows:

$$M = C_2 / e(C_1, sk_{A\_t}) \tag{3.8}$$

5) $RekeyGen(params, sk_{A\_t}, pk_{B\_t}) \rightarrow Rekey_{A \rightarrow B}$
After receiving the notification of successful authentication of Bob, Alice randomly generates an element $X \in G_T$ and computes the following partial re-encryption key:

$$Rk_1 = X \cdot e(g^s, H_1(ID_B) \cdot g^t)^r \tag{3.9}$$

$$Rk_2 = e\left(g^r, sk_{A\_t}^{-1} \cdot H_2(X)\right) \tag{3.10}$$

Output the partially ReKey as follows:

$$Rekey_{A \rightarrow B} = (Rk_1 \parallel Rk_2) \tag{3.11}$$

The $Rekey_{A \rightarrow B}$ is split and sent, only part of the rekey Rk_2 and ciphertext C are sent to the randomly selected encryption proxy nodes, and part of the rekey Rk_1 is split and sent to the other proxy nodes.

6) $ReEncrypt(params, C, Rekey_{A\to B}) \to C'$
Upon receiving the data, the encryption proxy node can encrypt C. The encryption process is as follows:

$$C_1' = C_1 \tag{3.12}$$

$$C_2' = C_2 \cdot Rk_2 \tag{3.13}$$

Output C_1' and C_2', and send them to Bob.

7) $KeyDecrypt(params, C', sk_B) \to M$
Bob concatenates the partial ReKey Rk_1 that has been received from the remaining proxy nodes to obtain the complete Rk_1, and sets $C_3' = Rk_1$. Then he combines the re-encrypted ciphertext C_1' and C_2' received from the re-encrypting proxy node to form the complete re-encrypted ciphertext $C' = C_1' \parallel C_2' \parallel C_3'$. Finally, he decrypts the complete re-encrypted ciphertext C' by using sk_{B_1} to obtain M.
The previously randomly generated element X can be computed by using the private key sk_{B_1},

$$X = C_3'/e(C_1', sk_{B\_1}) \tag{3.14}$$

Then M is obtained by using the previously generated element X:

$$M = C_2'/e\left(C_1', H_2\left(X\right)\right) \tag{3.15}$$

After decrypting to obtain M, Bob uses $SHA-256$ to hash it, consequently obtaining H_M'. Then, H_M' is compared with H_M. If $H_M' = H_M$, it indicates that the plaintext has not been tampered with or replaced, and C_{date} can be obtained by using the hash index H_{IPFS} in the IPFS. Finally, the EMRs can be obtained by decrypting it using *key* of the plaintext.
The scheme satisfies consistency, where for any set of public parameters, any plaintext M, and any user identity information, the above equation holds true.

3.4. Application scenarios

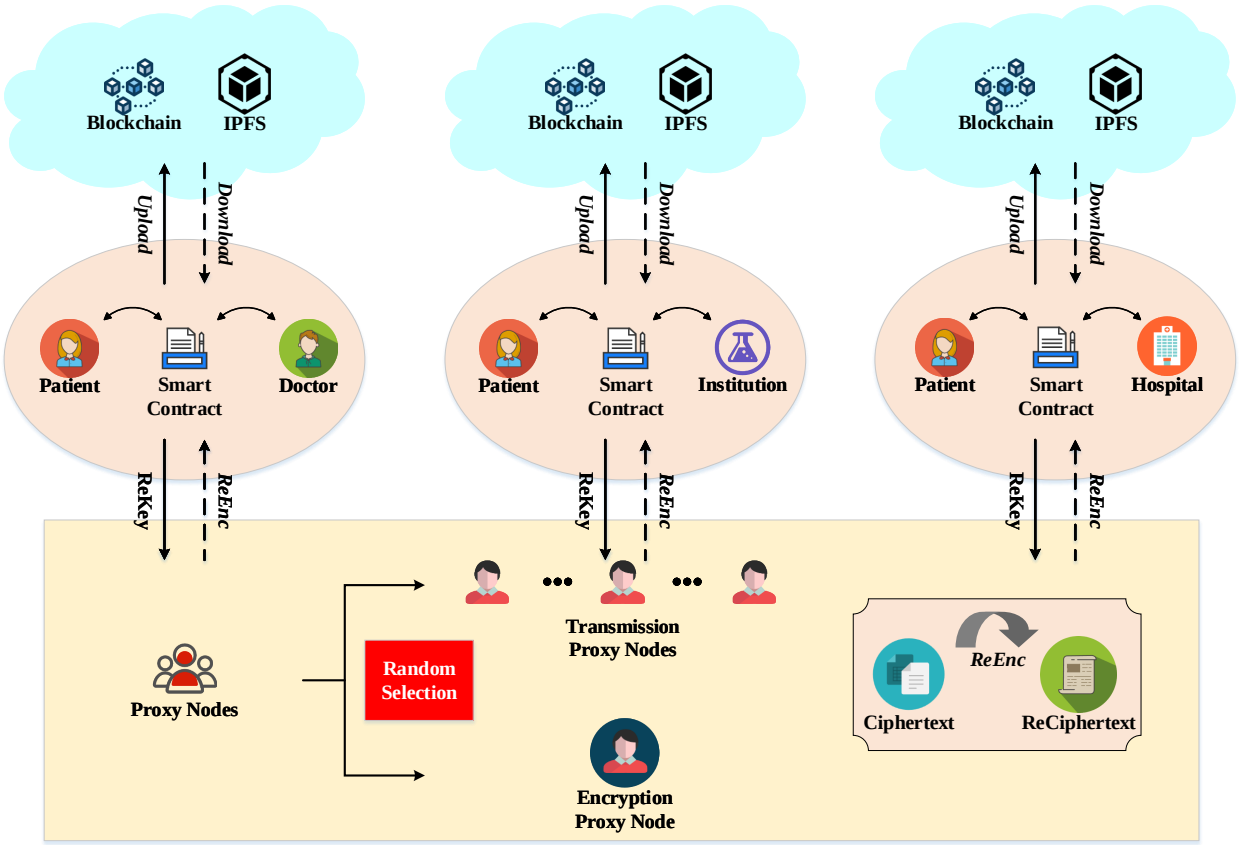

Figure 3. Conceptual diagram for the application scenario.

As shown in Figure 3, the conceptual diagram for the actual application scenario for this EMR sharing scheme is mainly concerned with the EMR sharing between patients and doctors, patients and scientific research institutions, and patients and hospitals. The shared data are uploaded and downloaded through the use of the blockchain and IPFS, and the two parties sharing the EMR realized access verification and generated a shared key through the use of the smart contracts. The proxy node cluster will select the encryption proxy node by implementing a random algorithm every predetermined interval, and perform proxy re-encryption by using part of the ReKey and ciphertext to obtain the re-ciphertext; the other proxy nodes will be used as transmission proxy nodes to send the other part of the ReKey. After receiving the complete information of the re-ciphertext, the data requester can decrypt the data through the use of the shared private key.

The smart contract process to achieve access verification is shown in Figure 4. The data owner Alice needs to initiate the sharing of EMRs through the system compilation smart contract. The content of the deployment contract includes access conditions such as the blockchain network partition to which it belongs, the allowed user's ID and the access time T_{limit}. Once the smart contract compilation is finalized, the contract generates a shared public key pk_1, which is specific to this medical record information sharing, based on the current timestamp T_{now} and the user's public key. This key is exclusively utilized for the purpose of sharing medical record information. When data

requester Bob requests access to the EMR from data owner Alice, it is subject to verification by the smart contract. After receiving the access application, the smart contract first determines whether its user identity satisfies the access conditions, whether the access time is within the access time, and whether it needs to pay the cross-zone fee. After completing the verification, the shared public key pk_{B_t} corresponding to user Bob is sent to Alice and Bob, otherwise, the smart contract informs that user Bob has no right to access. Upon receiving the pk_{B_t}, Bob can generate the shared private key sk_{B_t} through the use of the PKG. All transaction information of the contract will be stored in the blockchain system, and the data will not be subjected to tampering once uploaded.

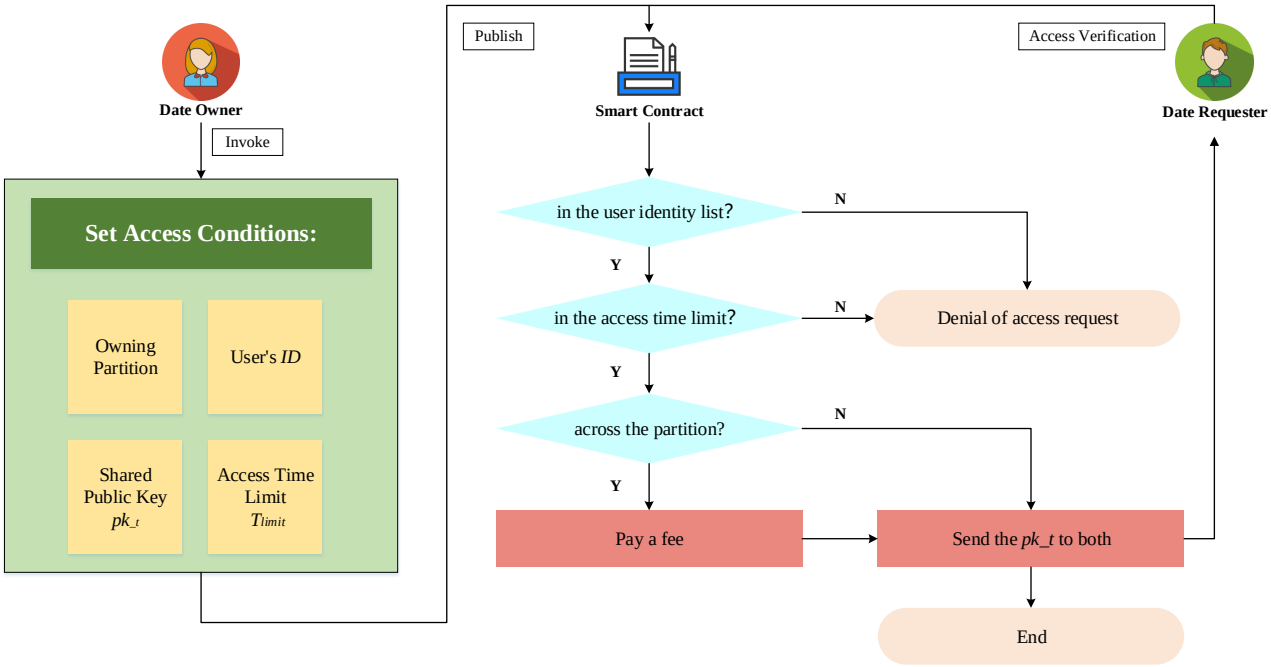

Figure 4. Access verification process.

4. Security proof

4.1. Chosen plaintext attack security

The security proof for this scheme is based on the IBPRE scheme, which implies that if the advantage of attacker $\mathcal{A}$ under the random oracle model can be neglected, then the scheme is chosen plaintext secure. To prove the chosen plaintext attack (CPA) security of the scheme, the following model has been designed.

Theorem If the IBPRE scheme has the goal of CPA security, meaning that there is no attacker $\mathcal{A}$ that can break the IBPRE scheme in polynomial time, then the scheme proposed in this paper also possesses CPA security.

Proof of Theorem 1 Suppose that there exists a probabilistic polynomial-time attacker $\mathcal{A}$ that can break the security of our proposed scheme with a non-negligible advantage; then, we can construct a challenger $\mathcal{B}$ who can break the IBPRE scheme.

Pre-preparation: $\mathcal{A}$ provides the user's ID^* and timestamp T^* that they want to challenge, and

sends them to $\mathcal{B}$.

Setup: $\mathcal{B}$ queries the public parameters $pars = (G_1, G_T, p, g, e, H_1, H_2)$ from the IBPRE scheme, selects a random number $s \in Z_p^*$ as the master secret key, computes g^s as the master public key, and finally outputs $params = (G_1, G_T, p, g, g^s, e, H_1, H_2)$ to $\mathcal{A}$.

Query 1: $\mathcal{A}$ can make the following two types of queries:

1) Query for the private key: If $\mathcal{A}$ provides a user's ID, and if $ID \neq ID^*$, $\mathcal{B}$ calculates $sk = H_1(ID)^s$ and returns it to $\mathcal{A}$ as the private key.

2) Query for ReKey: For the user's ID and timestamp T provided by $\mathcal{A}$, if $ID \neq ID^*$ and $T \neq T^*$, $\mathcal{B}$ computes the hash value of timestamp $T \in \{0,1\}^*$ and maps it to Z_p^*, denoted as $t \in Z_p^*$. Then, a random element $X \in G_T$ is generated, and the $Rekey_{A\to B} = (Rk_1 \| Rk_2)$ is computed as follows: $Rk_1 = X \cdot e(g^s, H_1(ID_B) \cdot g^t)^r$, and $Rk_2 = e\left(g^r, sk_{A\_t}^{-1} \cdot H_2(X)\right)$. Finally, the $Rekey_{A\to B}$ is returned to $\mathcal{A}$.

Challenge: $\mathcal{A}$ sends two equally-long encrypted messages M_0 and M_1 to $\mathcal{B}$. $\mathcal{B}$ randomly chooses $b \in \{0,1\}$, encrypts the message M_b to obtain the ciphertext C, and returns it to $\mathcal{A}$.

Query 2: $\mathcal{A}$ continues the key query process from Phase 1.

Guess: $\mathcal{A}$ outputs a guess value b' from the set $\{0,1\}$, and if $b' = b$, $\mathcal{A}$ wins the game.

As the model shows, in the re-encryption key query phase, $\mathcal{B}$ generates t as derived from the mapping of T, and there must exist an element $t' \in Z_p^*$ such that $t' = t$. For the attacker $\mathcal{A}$, the goal is to distinguish whether the obtained re-encryption key is generated from the t derived from the mapping or t' generated randomly, but t and t' have the exact same distribution. Therefore, the advantage of $\mathcal{A}$ in the game, denoted as ADV, is the same as that of the IBPRE scheme. Therefore, if $\mathcal{A}$ can successfully break the proposed scheme with advantage ADV, then $\mathcal{B}$ can also break the IBPRE scheme with advantage ADV, which contradicts the known security of the IBPRE scheme. Thus, the proposed scheme achieves chosen plaintext security.

4.2. Key security

The key security of this scheme mainly involves two components: the master key of the PKG and the private keys of the users. The scheme assumes that the PKG role is undertaken by a trustworthy authority; thus, the master key of the PKG can be considered to be effectively protected and will not be disclosed. Under the premise of ensuring that the master key is not leaked, as long as users can correctly use their private keys for offline generation, the security of their private keys can be guaranteed.

4.3. Anti-impersonation attacks security

According to the above, this scheme employs the generation of temporary shared keys and random task assignment to multiple proxy nodes during the proxy re-encryption process to mitigate impersonation attacks. According to Eqs (3.14) and (3.15), it is evident that malicious node Charles, in attempting to steal user privacy through impersonation attacks, would need to simultaneously acquire the temporary shared key $(pk_\_, sk_\_)$ and complete re-ciphertext $C' = C'_1 \| C'_2 \| C'_3$ to compute plaintext M. Therefore, the analysis of potential impersonation attacks in the EMR sharing process can be divided into the following four cases:

Cases1: The malicious node Charles itself is not a proxy node, and it can make Alice believe Charles to be a proxy node by means of impersonation or other means.

Through impersonation, Charles may obtain partial information that is required for proxy

re-encryption; but, based on key security, he cannot obtain the shared private key sk_{B_t} of data requester Bob. According to Eq (3.14), even if Charles possesses partial re-ciphertext C'_1 and C'_3, without knowledge of the shared private key sk_{B_t}, he is unable to compute the random element X. Consequently, he cannot further decipher plaintext M to access shared EMR information, so the attack fails.

Cases2: The malicious node Charles itself is not a proxy node, and it can make Alice believe that Charles is the data requesters Bob by means of impersonation or other means.

Through impersonation, Charles may obtain Bob's shared key (pk_{B_t}, sk_{B_t}) through multi-factor strong authentication, but the complete re-ciphertexts $C' = C'_1 \parallel C'_2 \parallel C'_3$ ultimately remains accepted by the genuine data requester Bob, inaccessible to Charles. According to Eq (3.14), it is evident that solely relying on the shared private key sk_{B_t} does not allow for computation of the random element X. Consequently, Charles cannot further decipher plaintext M to access shared EMR information, so the attack fails.

Cases3: The malicious node Charles itself is a proxy node and an encryption proxy node, and it can convince Alice that Charles is the data requesters Bob by means of impersonation or other means.

Through impersonation, Charles can obtain Bob's shared key (pk_{B_t}, sk_{B_t}) through multi-factor strong authentication, and acquire the partial re-ciphertexts C'_1 and C'_2 required by the encryption proxy nodes through random task assignment; however, he fails to obtain partial re-ciphertext C'_3. According to Eq (3.14), it is evident that the absence of partial re-ciphertext C'_3 prevents the computation of the random element X. Consequently, Charles cannot further decipher plaintext M to access shared EMR information, so the attack fails.

Cases4: The malicious node Charles itself is a proxy node and a transmission proxy node, and it can convince Alice that Charles is the data requesters Bob by means of impersonation or other means.

Through impersonation, Charles can acquire Bob's shared key (pk_{B_t}, sk_{B_t}) through multi-factor strong authentication and obtain partial re-ciphertext fragments, represented by C'_3 required by the transmission proxy nodes through random task assignment. Since C'_3 is divided into multiple fragments sent by various transmission proxy nodes to the genuine data requester Bob, Charles can only obtain the complete partial re-ciphertext C'_3 by collaborating with all transmission proxy nodes across the network. According to Eqs (3.14) and (3.15), even if Charles obtains the complete partial re-ciphertext C'_3 and computes the random element X, the absence of partial re-ciphertexts C'_1 and C'_2 prevents him from further deciphering plaintext M to access shared electronic medical record information, so the attack fails.

Based on the analysis above, the following conclusion can be drawn: For malicious node Charles to obtain both the temporary shared key (pk_{B_t}, sk_{B_t}) and the complete re-ciphertext $C' = C'_1 \parallel C'_2 \parallel C'_3$, he must jointly control all proxy nodes across the entire network, which would allow him to decrypt and access shared EMR information. However, in practical applications, controlling all proxy nodes that are trusted by participants is infeasible. Therefore, this scheme exhibits a high level of security against impersonation attacks.

4.4. Anti-collusion attack security

If the proxy node is colluding with the data requester, the element X can be calculated by knowing the ReKey and the shared private key sk_{B_t} of the data requester, and then the shared private key sk_{A_t} of the data owner can be theoretically calculated by inverting Rk_2. However, Rk_2 is obtained by bilinear

pairing operation, and its inverse operation involves computing the discrete logarithm problem, which is a known hard mathematical problem. In addition, even if the attacker can calculate the shared private key sk_{A_t} of the data owner by using a large amount of computing power, according to the above scheme definition, the shared private key sk_{A_t} is only used to encrypt the shared M and has no other purpose. At the same time, based on the key security, the system master key is effectively protected and will not be leaked, and the shared private key sk_{A_t} can not be used to obtain the user private key of the data owner through calculation, so the scheme is secure against collusion attack.

5. Scheme analysis

5.1. Comparison of schemes

Table 2 presents a comparison of the IBPRE scheme based on blockchain and identity proxy re-encryption with similar data security sharing schemes. The scheme developed by Xu et al. [34] fails to effectively counter collusion and impersonation attacks. The scheme developed by Yao et al. [35] can realize the real-time update of the encryption key, but it still has vulnerabilities in dealing with impersonation attacks. The scheme developed by Kan et al. [36] solves the difficulty that the traditional data deduplication technology cannot eliminate the duplication of encrypted data, but the cloud server still faces the risk of insufficient storage space. In this scheme, the IPFS distributed storage is used to solve the problems faced by cloud storage; then, the user's access control to data is realized through the use of smart contracts. In the sharing process, the scheme generates the shared key pair ($pk_{\_t}$, $sk_{\_t}$) based on the timestamp T_{now}, and circumvents the hidden danger caused by impersonation and collusion attacks by applying the task allocation of multiple proxy nodes. Finally, after the user obtains the plaintext data, it is verified by the blockchain network according to the hash value to ensure the integrity of the data.

Table 2. Functional comparison of related schemes.

Function	Scheme [34]	Scheme [35]	Scheme [36]	Our Scheme
Distributed data storage	×	×	×	✓
Data access control	✓	✓	✓	✓
Integrity check	×	×	✓	✓
Anti-collusion attack	×	✓	✓	✓
Anti-impersonation attacks	×	×	✓	✓

The scheme measures computational costs by using bilinear and exponential operations, as they are relatively time-consuming compared to hash operations and non-generator group exponential operations, which are negligible. t_e represents one exponential operation on a cyclic group, and t_p represents one bilinear operation; "-" represents no bilinear operation or exponential operation. The scheme has been compared with the schemes proposed in [34–36], and the results are shown in Table 3.

Table 3. Calculation cost comparison for related schemes.

Calculation Cost	Scheme [34]	Scheme [35]	Scheme [36]	Our Scheme
Enc	$3t_e + t_p$	$8t_e + t_p$	$t_e + 2t_p$	$2t_e + t_p$
ReEnc	$4t_e + t_p$	$t_e + 2t_p$	t_p	-
Dnc	$t_e + 3t_p$	$t_e + t_p$	$3t_p$	$2t_p$
Rekey	$4t_e$	$6t_e$	$2t_e + t_p$	$t_e + 2t_p$

5.2. Experiments

The hardware environment for this experiment consisted of a server equipped with an Intel Xeon Ice Lake processor running at a base frequency of 2.7 GHz, with 8 cores CPU, and 32GB RAM. The server ran a CentOS 7.6 operating system. Additionally, all applications implemented in this experiment were deployed as containers by using Docker software. The identity-based proxy re-encryption algorithm was implemented by using the Java Pairing Based Cryptography (JPBC) library, with type A curves and parameter settings of r = 256 and q = 512. The SHA-256 algorithm was used for hashing. The local blockchain environment was configured by using the open-source project Chain33 source code, with modifications made to run and complete the experimental tests.

5.2.1. Blockchain performance testing

First, a performance comparison of the blockchain platforms used was conducted. The experimental versions of the blockchain platforms were Hyperledger Fabric 1.4.0, Tendermint 0.34.0, and Chain33 1.65.0. The network consensus nodes of Hyperledger Fabric, Tendermint, and Chain33 blockchain platforms were uniformly set to 4. Two organizations, four peer nodes, one orderer node, and one CA node were deployed using Docker. The experimental tests were conducted by setting concurrent request numbers of 200, 600, 1000, 2000, and 3000. The performance comparison of different blockchain platforms is shown in Figure 5. From Figure 5, it can be observed that at lower concurrent request numbers, the differences between the three blockchain platforms are small. However, as the concurrent request numbers increase, the overall transaction performance of the Chain33 platform surpasses that of the other two platforms.

Subsequently, the transaction processing performance of Chain33 was tested by deploying networks with 4, 6, 8, 10, and 12 validator nodes to assess the performance variation of blockchain networks with different node counts. In this experiment, different levels of concurrent request loads were set at 200, 600, 1000, 2000, and 3000 transactions per second (TPS). The experimental results for transaction processing performance in the local blockchain environment are presented in Figure 6.

From Figure 6, it can be observed that as the number of concurrent requests increases, the performance of blockchain networks with different node counts tends to decrease. This is primarily due to the impact of network transmission rates. Overall, as concurrency increases, the blockchain network demonstrates stable performance without timeouts. Regarding the performance of networks with different numbers of nodes under the same concurrency level, there is a general trend of decreasing TPS as the number of nodes increases. For example, when the concurrent request count is 2000, the TPS of the blockchain network gradually decreases as the number of nodes increases from 4 to 12 nodes. As the number of nodes in the network increases, communication overhead gradually

increases, which in turn affects the overall blockchain performance.

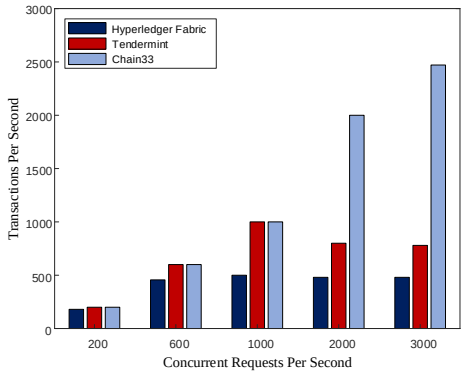

Figure 5. Blockchain platform performance comparison.

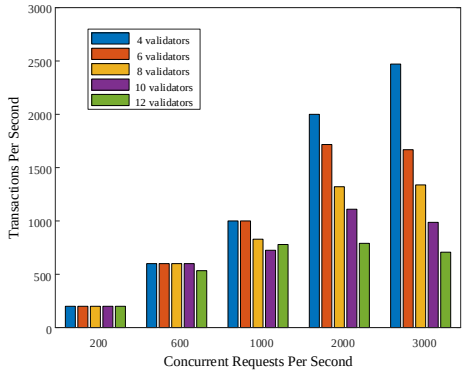

Figure 6. Transaction processing performance comparison.

5.2.2. Proxy re-encryption performance testing

In this experiment, file encryption was performed by using the Advanced Encryption Standard (AES) algorithm with a 128-bit key length. The time required for this operation depends on the file size and is not highly correlated with the identity-based proxy re-encryption algorithm itself. Therefore, this experiment does not include testing for file encryption time. At the same time, based on the theory of this study, it can be known that the defined plaintext of the scheme is always a fixed length, and the time cost of proxy re-encryption in the scheme is theoretically a fixed value. Therefore, the time costs of Rekey generation and proxy re-encryption have been compared, and the time of Rekey generation or re-encryption is compared by testing 5, 10, 20, 50, and 100 times respectively. In order to ensure the accuracy of the test data, the time consumption of each test item was tested 100 times and the average value was taken; the final results are shown in Figures 7 and 8 below.

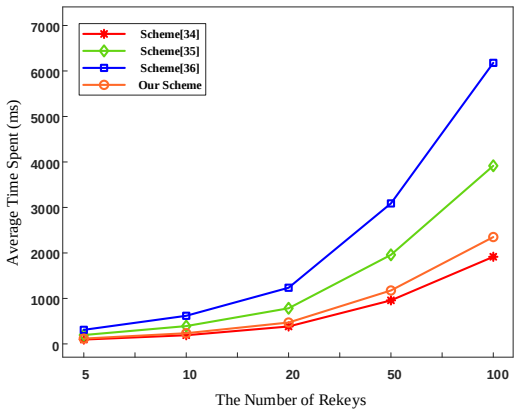

Figure 7. Comparison of ReKey generation time.

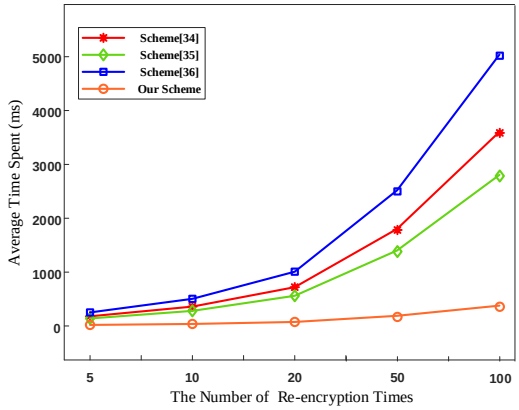

Figure 8. Comparison of reencryption time.

From Figure 7, it can be seen that when the number of generated re-encryption keys is relatively low, our scheme performs similarly to those of [34–36]. However, by increasing the number of generated re-encryption keys, we found that our scheme's average time for generating re-encryption keys is slightly longer than that of the scheme of [34], but it exhibited better performance than the schemes of [35, 36]. From Figure 8, it can be observed that when the re-encryption frequency is low, our scheme has a slight advantage over the schemes of [34–36], and as the re-encryption frequency increases, our scheme's average re-encryption time relative to the schemes of [34–36], exhibits better performance. In summary, considering the sharing process of our scheme, although the average time taken by our scheme to generate re-encryption keys is slightly longer than that of the scheme of [34], the frequency of generating re-encryption keys in the actual sharing process is relatively low and does not significantly affect the overall sharing scheme flow. However, the advantage of our scheme in terms of re-encryption contributes to further enhancing the efficiency of the sharing scheme. Additionally, the scheme of [34] performs poorly in terms of resisting malicious attacks, while our scheme can provide stronger security. Therefore, our scheme can offer a more secure sharing environment while ensuring high efficiency.

6. Conclusions

This scheme eliminates the potential threat of an impersonation attack by temporarily sharing the key and randomly assigning multiple proxy node tasks. A random function was used to ensure the fairness of task allocation of proxy nodes. The method of the IPFS to store ciphertext and index information on the blockchain solved the problems of a single point of failure in centralized management and blockchain storage overload. The smart contract has been used to realize the shared EMR access control of the data requesters, and the shared key is generated according to the timestamp. Finally, security analysis and experimental tests show that the scheme enables satisfactory levels of CPA security, key security, anti-impersonation attack security and anti-collusion attack security, as well as realizes privacy protection and an efficiency guarantee in the sharing process ; furthermore, it can be applied to the sharing scenario of EMRs. In the future, the research on the security of EMR sharing can further combine game theory [37–39], optimization [40–42] and artificial intelligence methods, such as broad learning [43–45] and unsupervised learning [46], to provide more efficient and secure services for the medical field.

Use of AI tools declaration

The authors declare that they have not used artificial intelligence tools in the creation of this article.

Acknowledgments

The authors are grateful for the support from the 2020 Ministry and Province Joint Construction Fund 605-52520005, the High-Level Talent Innovation and Entrepreneurship Project 2022C020R, Fujian Provincial Guiding Project 2023H0012, the Scientific Research Funds of Huaqiao University under no. 21BS128 and the Natural Science Foundation of Fujian Province under no. 2022J05065.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. A. ElShafee, W. El-Shafai, Design and analysis of data link impersonation attack for wired LAN application layer services, *J. Amb. Intell. Human. Comput.*, **14** (2023), 13465–13488. https://doi.org/10.1007/s12652-022-03800-5
2. K. Yang, Y. Shi, Z. Yu, Q. Yang, A. Sangaiah, H. Zeng, Stacked one-class broad learning system for intrusion detection in industry 4.0, *IEEE Trans. Industr. Inform.*, **19** (2023), 251–260. https://doi.org/10.1109/TII.2022.3157727
3. K. Yang, Z. Yu, C. Chen, W. Cao, H. Wong, J. You, et al., Progressive hybrid classifier ensemble for imbalanced data, *IEEE Trans. Syst. Man Cybern. Syst.*, **52** (2022), 2464–2478. https://doi.org/10.1109/TSMC.2021.3051138
4. L. Guo, W. Gao, Y. Cao, X. Lai, Research on medical data security sharing scheme based on homomorphic encryption, *Math. Biosci. Eng.*, **20** (2023), 2261–2279. https://doi.org/10.3934/mbe.2023106
5. Y. Lu, D. Zhao, An anonymous SIP authenticated key agreement protocol based on elliptic curve cryptography, *Math. Biosci. Eng.*, **19** (2022), 66–85. https://doi.org/10.3934/mbe.2022003
6. J. Xu, Y. Tian, T. Ma, N. Al-Nabhan, Intelligent manufacturing security model based on improved blockchain, *Math. Biosci. Eng.*, **17** (2020), 5633–5650. https://doi.org/10.3934/mbe.2020303
7. A. Zakzouk, A. El-Sayed, E. Hemdan, A blockchain-based electronic medical records management framework in smart healthcare infrastructure, *Mult. Tools Appl.*, **82** (2023), 35419–35437. https://doi.org/10.1007/s11042-023-15152-z
8. W. Wang, D. Teng, M. Chen, Y. Ge, Y. Zou, A trading matching model for aquatic products based on blockchain and credit mechanisms, *Math. Biosci. Eng.*, **20** (2023), 19732–19762. https://doi.org/10.3934/mbe.2023874
9. M. Du, Q. Chen, J. Chen, X. Ma, An optimized consortium blockchain for medical information sharing, *IEEE Trans. Eng. Manag.*, **68** (2020), 1677–1689. https://doi.org/10.1109/TEM.2020.2966832
10. K. Neela, V. Kavitha, Blockchain based chaotic deep gan encryption scheme for securing medical images in a cloud environment, *Appl. Intell.*, **53** (2023), 4733–4747. https://doi.org/10.1007/s10489-022-03730-x
11. J. Qu, Blockchain in medical informatics, *J. Industr. Inform. Integr.*, **25** (2022), 100258. https://doi.org/10.1016/j.jii.2021.100258
12. H. Huang, P. Zhu, F. Xiao, X. Sun, Q. Huang, A blockchain-based scheme for privacypreserving and secure sharing of medical data, *Comput. Secur.*, **99** (2020), 102010. https://doi.org/10.1016/j.cose.2020.102010
13. J. Liu, W. Jiang, R. Sun, A. Bashiret, M. Alshehri, Q. Hua, et al., Conditional anonymous remote healthcare data sharing over blockchain, *IEEE J. Biomed. Health Inform.*, **27** (2022), 2231–2242. https://doi.org/10.1109/JBHI.2022.3183397
14. M. Wang, Y. Guo, C. Zhang, C. Wang, H. Huang, X. Jia, Medshare: A privacy-preserving medical data sharing system by using blockchain, *IEEE Trans. Serv. Comput.*, **16** (2021). https://doi.org/10.1109/TSC.2021.3114719

15. V. Marichamy, V. Natarajan, Blockchain based securing medical records in big data analytics, *Data Knowl. Eng.*, **14** (2023), 102122. https://doi.org/10.1016/j.datak.2022.102122

16. X. Yang, W. Wang, T. Tian, C. Wang, Cryptanalysis and improvement of a blockchain-based certificateless signature for IIoT devices, *IEEE Trans. Industr. Inform.*, **20** (2024), 1884–1894. https://doi.org/10.1109/TII.2023.3282317

17. X. Yang, H. Wen, R. Diao, X. Du, C. Wang, Improved security of a pairing-free certificateless aggregate signature in healthcare wireless medical sensor networks, *IEEE Int. Things J.*, **10** (2023), 10881–10892. https://doi.org/10.1109/JIOT.2023.3240426

18. A. Hisham, Detecting impersonation attacks in cloud computing environments using a centric user profiling approach, *Future Gener. Comput. Syst.*, **117** (2021), 299–320. https://doi.org/10.1016/j.future.2020.12.009

19. X. Yang, S. Li, M. Li, X. Du, C. Wang, Heterogeneous signcryption scheme from PKI to IBC with multi-ciphertext equality test in internet of vehicles, *IEEE Int. Things J.*, (2023), 1. https://doi.org/10.1109/JIOT.2023.3341146

20. M. Ma, M. Luo, S. Fan, D. Feng, An efficient pairing-free certificateless cearchable public key encryption for cloud-based IIoT, *Wirel. Commun. Mobile Comput.*, 2020. https://doi.org/10.1155/2020/8850520

21. Z. Zhang, S. Zhou, A decentralized strongly secure attribute-based encryption and authentication scheme for distributed Internet of Mobile Things, *Comput. Networks*, **201** (2021), 108553. https://doi.org/10.1016/j.comnet.2021.108553

22. M. Blaze, G. Bleumer, M. Strauss, Divertible protocols and atomic proxy cryptography, *Int. Confer. Theory Appl. Cryptographic Techn.*, (1998), 127–144. https://doi.org/10.1007/BFb0054122

23. H. Guo, Z. Zhang, J. Xu, N. An, X. Lan, Accountable proxy re-encryption for secure data sharing, *IEEE Trans. Depend. Secure Comput.*, **18** (2018), 145–159. https://doi.org/10.1109/TDSC.2018.2877601

24. C. Fan, J. Chen, S. Huang, J. Huang, W. Chen, Provably secure timed-release proxy conditional reencryption, *IEEE Syst. J.*, **11** (2015), 2291–2302. https://doi.org/10.1109/JSYST.2014.2385778

25. K. Azbeg, O. Ouchetto, S. Andaloussi, Blockmedcare: A healthcare system based on iot, blockchain and ipfs for data management security, *Egypt. Inform. J.*, **23** (2022), 320–343. https://doi.org/10.1016/j.eij.2022.02.004

26. A. Manzoor, A. Braeken, S. Kanhere, M. Ylianttila, M. Liyanage, Proxy re-encryption enabled secure and anonymous iot data sharing platform based on blockchain, *J. Network Comput. Appl.*, **176** (2021), 102917. https://doi.org/10.1016/j.jnca.2020.102917

27. M. Green, G. Ateniese, Identity-Based proxy re-encryption, *Appl. Cryptography Network Secur.*, **4521** (2007), 288–306. https://doi.org/10.1007/978-3-540-72738-5_19

28. H. Wang, Z. Cao, L. Wang, Multi-use and unidirectional identity-based proxy re-encryption schemes, *Inform. Sci.*, **180** (2010), 4042–4059. https://doi.org/10.1016/j.ins.2010.06.029

29. H. Xiong, L. Wang, Z. Zhou, Z. Zhao, X. Huang, S. Kumari, Burn after reading: Adaptively secure puncturable identity-based proxy re-encryption scheme for securing group message, *IEEE Int. Things J.*, **9** (2021), 11248–11260. https://doi.org/10.1109/JIOT.2021.3126230

30. S. Maiti, S. Misra, P2b: Privacy preserving identity-based broadcast proxy re-encryption, *IEEE Trans. Veh. Technol.*, **69** (2020), 5610–5617. https://doi.org/10.1109/TVT.2020.2982422

31. C. Ge, Z. Liu, J. Xia, L. Fang, Revocable identity-based broadcast proxy reencryption for data sharing in clouds, *IEEE Trans. Depend. Secure Comput.*, **18** (2019), 1214–1226. https://doi.org/10.1109/TDSC.2019.2899300

32. Y. Zhou, L. Zhao, Y. Jin, F. Li, Backdoor-resistant identity-based proxy reencryption for cloud-assisted wireless body area networks, *Inform. Sci.*, **604** (2022), 80–96. https://doi.org/10.1016/j.ins.2022.05.007

33. H. Lin, T. Tsai, P. Ting, Y. Fan, Identity-based proxy re-encryption scheme using fog computing and anonymous key generation, *Sensors*, **23** (2023), 2706. https://doi.org/10.3390/s23052706

34. P. Xu, T. Jiao, Q. Wu, W. Wang, H. Jin, Conditional identity-based broadcast proxy reencryption and its application to cloud email, *IEEE Trans. Comput.*, **65** (2015), 66–79. https://doi.org/10.1109/TC.2015.2417544

35. S. Yao, R. Dayot, H. Kim, I. Ra, A novel revocable and identity-based conditional proxy re-encryption scheme with ciphertext evolution for secure cloud data sharing, *IEEE Access*, **9** (2021), 42801–42816. https://doi.org/10.1109/ACCESS.2021.3064863

36. G, Kan, C. Jin, H. Zhu, Y. Xu, N. Liu, An identity-based proxy re-encryption for data deduplication in cloud, *J. Syst. Arch.*, **121** (2021), 102332. https://doi.org/10.1016/j.sysarc.2021.102332

37. J. Bi, F. Luo, S. He, G. Liang, W. Meng, M. Sun, False data injection- and propagation-aware game theoretical approach for microgrids, *IEEE Trans. Smart Grid*, **13** (2022), 3342–3353. https://doi.org/10.1109/TSG.2022.3174918

38. J. Bi, S. He, F. Luo, W. Meng, L. Ji, D. Huang, Defense of advanced persistent threat on industrial internet of things with lateral movement modelling, *IEEE Trans. Industr. Inform.*, **19** (2023), 9619–9630. https://doi.org/10.1109/TII.2022.3231406

39. J. Bi, S. He, F. Luo, J. Chen, D. Huang, M. Sun, Differential game approach for modelling and defense of false data injection attacks targeting energy metering systems, in *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, (2022), 97–104. https://doi.org/10.1109/TrustCom56396.2022.00024

40. J. Bi, F. Zhang, A. Dorri, C. Zhang, C. Zhang, A risk management approach to double-virus tradeoff problem, *IEEE Access*, **7** (2019), 144472–144480. https://doi.org/10.1109/ACCESS.2019.2944985

41. J. Bi, F. Luo, G. Liang, X. Yang, S. He, Z. Dong, Impact assessment and defense for smart grids with FDIA against AMI, *IEEE Trans. Network Sci. Eng.*, **10** (2022), 578–591. https://doi.org/10.1109/TNSE.2022.3197682

42. D. Huang, F. Luo, J. Bi, M. Sun, An efficient hybrid IDS deployment architecture for multi-hop clustered wireless sensor networks, *IEEE Trans. Inform. Forens. Secur.*, **17** (2022), 2688–2702. https://doi.org/10.1109/TIFS.2022.3191491

43. Y. Shi, K. Yang, Z. Yu, C. Chen, H. Zeng, Adaptive ensemble clustering with boosting BLS-based autoencoder, *IEEE Trans. Knowl. Data Eng.*, **35** (2023), 12369–12383. https://doi.org/10.1109/TKDE.2023.3271120

44. K. Yang, Z. Yu, C. Chen, W. Cao, J. You, H. Wong, Incremental weighted ensemble broad learning system for imbalanced data, *IEEE Trans. Knowl. Data Eng.*, **12** (2022), 5809–5824. https://doi.org/10.1109/TKDE.2021.3061428

45. K. Yang, Y. Liu, Z. Yu, C. Chen, Extracting and composing robust features with broad learning system, *IEEE Trans. Knowl. Data Eng.*, **35** (2023), 3885–3896. https://doi.org/10.1109/TKDE.2021.3137792

46. Y. Shi, Z. Yu, C. Chen, H. Zeng, Consensus clustering with co-association matrix optimization, *IEEE Trans. Neural Networks Learn. Syst.*, (2022), 1–14. https://doi.org/10.1109/TNNLS.2022.3201975

AIMS Press