# Conditional Anonymous Remote Healthcare Data Sharing Over Blockchain

Jingwei Liu, *Member, IEEE,* Weiyang Jiang, Rong Sun, *Member, IEEE,* Ali Kashif Bashir, *Senior Member, IEEE,* Mohammad Dahman Alshehri, *Member, IEEE,* Qiaozhi Hua, Keping Yu, *Member, IEEE*

*Abstract*— As an important carrier of healthcare data, Electronic Medical Records (EMRs) generated from various sensors, i.e., wearable, implantable, are extremely valuable research materials for artificial intelligence and machine learning. The efficient circulation of EMRs can improve remote medical services and promote the development of the related healthcare industry. However, in traditional centralized data sharing architectures, the balance between privacy and traceability still cannot be well handled. To address the issue that malicious users cannot be locked in the fully anonymous sharing schemes, we propose a trackable anonymous remote healthcare data storing and sharing scheme over decentralized consortium blockchain. Through an "on-chain & off-chain" model, it relieves the massive data storage pressure of medical blockchain. By introducing an improved proxy re-encryption mechanism, the proposed scheme realizes the fine-gained access control of the outsourced data, and can also prevent the collusion between semi-trusted cloud servers and data requesters who try to reveal EMRs without authorization. Compared with the existing schemes, our solution can provide a lower computational overhead in repeated EMRs sharing, resulting in a more efficient overall performance.

*Index Terms*— Remote medical data sharing, privacy preservation, blockchain, proxy re-encryption

## I. INTRODUCTION

WITH the rapid development of the Internet [1], [2] and big data technologies [3], society has entered an era of information and data sharing. Data has turned to the most valuable asset in this era. With the help of innovative technologies [4], [5], the healthcare industry is also shifting from traditional medicine to digital driven medicine. Healthcare data from different patients is very essential research material for improving public health. As a kind of important carrier of healthcare data, Electronic Medical Records (EMRs) can be widely used in data analysis and medical research. Compared with traditional physical medical records, EMRs have the advantages of reducing overhead, improving efficiency and mobility, facilitating storage and promoting the development of medical research [6], [7]. In 2009, the president of the United States signed "The American Recovery and Reinvestment Act" which included an investment of up to 19 billion dollars for the digitization of medical records [8].

During the transition from traditional physical medical records to EMRs, EMRs have raised many data security and privacy issues [9] due to the vulnerability to hackers, viruses and other threats. In recent years, there have been numerous incidents of healthcare data being lost and stolen. EMRs, in general, contain lots of sensitive information in patients' healthcare data. Therefore, how to ensure the data security and privacy preservation in sharing EMRs has become a crucial issue [10]. To avoid privacy leakage, traditional medical institutions usually build their own medical system servers to save patients' EMRs in [11], [12]. However, these servers often have many defects such as centralization, poor interactivity and high overhead.

In 2008, Satoshi Nakamoto proposed an electronic cash system based on peer-to-peer network [13]. Since then, as a revolutionary technology, blockchain has become a research hotspot in industrial and academic fields. In blockchain, blocks are connected together in chronological order to form a chained data structure through cryptographic algorithms, in which data can not be tampered with. By virtue of distributed networks and consensus mechanisms, blockchain ensures that the data on the chain can be verified by all nodes. Therefore, blockchain is fault-tolerant and immune to single point attacks, and especially suitable for digital cryptocurrency, finance, healthcare, credit investigation, e-government and other fields in depth. Blockchain provides a foundation of trust in untrusted environments and enable participants to collaborate more securely, reliably and efficiently. But the massive growth of data gradually causes the storage bottleneck of blockchain. According to [14], healthcare data will reach 25000 PB soon. To solve the storage issue, it is a feasible technical route

Jingwei Liu, Weiyang Jiang and Rong Sun are with the Shaanxi Key Laboratory of Blockchain and Secure Computing, Xidian University, Xi'an, 710071, China (e-mail: jwliu@mail.xidian.edu.cn, wyjiang@stu.xidian.edu.cn, rsun@mail.xidian.edu.cn).

Ali Kashif Bashir is with the Department of Computing and Mathematics, Manchester Metropolitan University, Manchester, M15 6BH, United Kingdom (e-mail: dr.alikashif.b@ieee.org).

Mohammad Dahman Alshehri is with the Department of Computer Science, College of Computers and Information Technology, Taif University, Taif, 21944, Saudi Arabia (e-mail: alshehri@tu.edu.sa).

Qiaozhi Hua is with the Computer School, Hubei University of Arts and Science, Xiangyang, 441000, China (e-mail: 11722@hbuas.edu.cn).

Keping Yu is with the Graduate School of Science and Engineering, Hosei University, Tokyo, 184-8584, Japan (e-mail: keping.yu@ieee.org).

of combining blockchain with cloud services. In [15], cloud computing virtualizes computing power and storage resources into services for different users in an on-demand delivery model. The nearly unlimited computing power and storage space provided by cloud services can easily overcome the disadvantages of traditional medical data storing systems such as high deployment overhead, limited computing power and inadequate storage resources.

Therefore, to meet the security and privacy preserving requirements in healthcare data sharing scenarios, we propose a conditional anonymous remote healthcare data sharing scheme over blockchain. The main contributions of this work can be summarized as follows:

1) We propose an anonymous and traceable authentication protocol that can not only protect users' identity privacy using pseudo identities but also reveal the identities of malicious nodes in certain conditions. Moreover, the protocol designs a batch verification method to improve the efficiency of verification.

2) To mitigate the tense shortage of healthcare data sharing, we propose a remote healthcare data sharing scheme based on an on-chain/off-chain model that greatly relieves on-chain storage stress by migrating massive data to cloud servers.

3) In the theoretical analysis on security, the proposed scheme meets all given security features, especially decentralization, identity tracing and anti-collusion security. In terms of performance simulation, by moving complex operations to the encryption stage that only runs once, the proposed scheme greatly reduces the computational overhead of the EMRs sharing stage including re-encryption key generation, re-encryption and decryption. As the number of sharing increases, the total computational overhead of our scheme is much lower than other schemes.

The rest of this paper is organized as follows. Section II introduces the related works. In section III, we formalize the system model, security requirements, and then introduce the relevant preliminaries used in our scheme. Section IV describes the proposed scheme in detail. Section V analyzes the security, correctness and functional features. Section VI carries on the performance analysis through simulation and comparison. Section VII summarizes this paper.

## II. RELATED WORKS

In order to address the security and privacy issues in healthcare data storing and sharing, researchers have already done a great deal of excellent works. Zhang et al. [16] put forward a controllable EMRs access scheme that supported diverse queries. Cao et al. [17] proposed a blockchain-based electronic health system, which could resist impersonation attacks and ensure that patients' EMRs were not tampered with or forged. Subsequently, Liu et al. [18] proposed a healthcare data sharing scheme named BPDS which adopted the ciphertext-policy attribute-based encryption (CP-ABE) to achieve fine-grained controllable access of EMRs. The scheme used content extraction signature to avoid privacy leakage. In

addition, Liu et al. [19] put forward a fine-grained controllable files access scheme based on blockchain and cloud services. Furthermore, there are many other excellent healthcare data sharing schemes such as [20]–[25]. In order to prevent a non-fully trusted cloud server from forging access logs and seizing data privacy, Noh et al. [26] proposed a patient-centered EMRs management system, in which users' access activities were stored in logs and anonymous identifiers were locked on blockchain to protect patients' identity privacy. Although this scheme utilized the proxy re-encryption scheme in [27], it can not resist the collusion between cloud servers and requestors to recover the private keys of data owners. In 2021, Feng et al. [28] proposed a secure and efficient data sharing scheme based on blockchain. The scheme deployed attribute-based encryption (ABE) for data access control through smart contracts. In [29], a blockchain-based data sharing scheme was put forward for tracing maliciously modified data, in which the original data and transaction data were respectively stored on two different blockchains. In 2019, Eltayieb et al. [30] designed a controllable access scheme for data outsourcing computing. The scheme was based on a certificateless proxy re-encryption method with access authentication. In 2022, Sun et al. [31] proposed an EMRs search scheme based on ElGamal blind signature, which could safely invoke a patient's previous EMRs while protecting his/her privacy and EMRs database. In [32], Thwin et al. proposed a controllable access scheme of EMRs that was still based on proxy re-encryption scheme, in which the semi-trusted cloud server could obtain the identities of all users during the registration stage. It was an insecure centralized identity management mechanism. In [33], Liu et al. put forward a healthcare data sharing scheme that could resist collusion attacks with a trusted third party. Tan et al. [34] put forward a secure and privacy-preserving EMRs sharing scheme based on blockchain, which took advantage of CP-ABE and blockchain to achieve traceability and direct revocation of COVID-19 EMRs.

## III. METHODOLOGY

### A. System Model

In healthcare data sharing scenarios, patients take part in the healthcare data sharing system to share their EMRs with authorized data requestors. However, EMRs are highly sensitive data with patients' privacy, so patients may not be willing to disclose their personal information in the process of data sharing. In order to protect the identity privacy of participants, we propose a conditional anonymous remote healthcare data sharing scheme. As shown in Fig. 1, medical consortium blockchain enables healthcare data to be circulated between patients and data requestors. The system model consists of two main components: Medical Consortium Blockchain (MCB) and Cloud Server (CS).

*1) Medical Consortium Blockchain (MCB):* The consortium blockchain network is composed of the following three types of nodes.

- User Node (UN): UNs are composed of data owners and data requestors. Data owners are the users who hold EMRs and are willing to share them with others in the
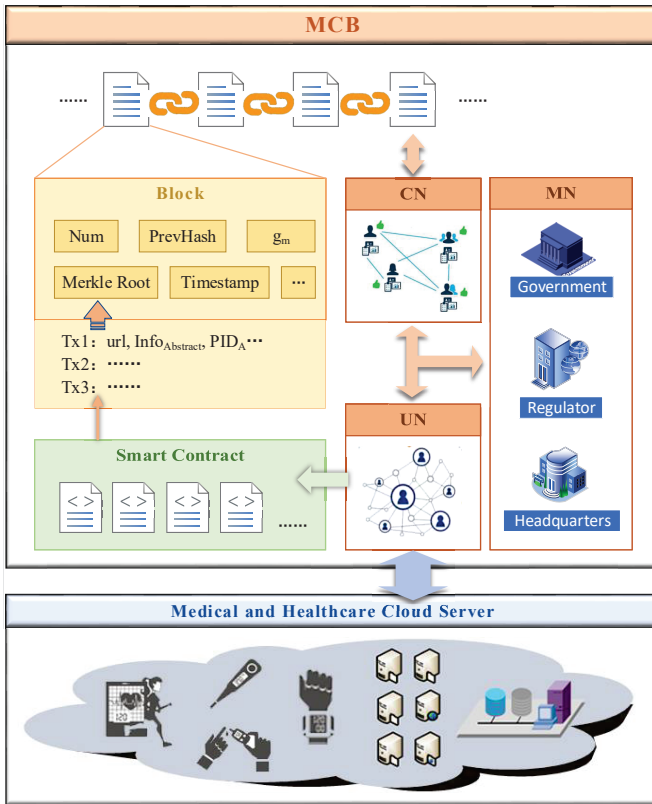
Fig. 1: System model

system, such as patients. Data requestors are the users who need to apply to data owners for accessing to EMRs. Data requestors are usually medical insurance companies or medical researchers. UN might be a data owner or a data requestor according to different businesses. Every UN can obtain and synchronize data on blockchain.

- Consensus Node (CN): CNs are the nodes that participate in the consensus process. They are in charge of generating and verifying data and blocks. In particular, CNs are responsible for registering and conditionally tracking the identities of UNs. They are usually authoritative institutions such as major hospitals, medical departments of universities and research institutions. In the consensus mechanism, CNs are divided into two roles: *Leader* and *Follower*.

- Management Node (MN): MN is usually held by the government department in charge of healthcare or the committee of the medical alliance organization. It plays a supervisory role and manages UNs' identity information.

*2) Cloud Server (CS):* As a semi-trusted third party, CS is primarily in charge of storing EMRs.

In this model, MN first generates the public system parameters. MN, CNs and CS independently select their own private keys and calculate the corresponding public keys. When a UN is joining the system, it needs to select a random number to cover its real identity, which forms identity protection information. Then, the UN shares the random number with all CNs through the Shamir secret sharing scheme. Each CN needs to validate the received share. If the verification passes, it sends a confirmation message to MN. Once collecting all confirmation messages, MN calculates and returns the corresponding

pseudo identity to the UN. Meanwhile, MN binds the identity protection information of the UN to the pseudonym as the tracking information and records it on the blockchain. Assume that Alice has joined the system and completed the registration process, she encrypts $EMR_A$ generated from diagnosis and sends it to CS who needs to verify $EMR_A$ and return the corresponding download address. Alice takes the abstract of $EMR_A$, the download address and her pseudo identity as metadata, which is verified and uploaded to the blockchain by CNs. When Bob wants to access $EMR_A$, he first retrieves the relevant information of $EMR_A$ on the blockchain and sends a request to Alice. If Alice allows Bob to access $EMR_A$, she generates a re-encryption key using Bob's request information and sends this key to CS. Then, CS converts the ciphertext of $EMR_A$ via the received re-encryption key. Now, Bob can download and decrypt the ciphertext using his own private key to obtain $EMR_A$. Finally, if UN is found to have malicious behavior, CNs can contribute their secret shares to recover the secret value of it by Lagrange interpolation polynomial, so as to further recover the real identity of the malicious UN. The identity tracking process is deployed as a smart contract that will execute automatically once certain conditions are met. The main symbols used in our scheme are indicated in TABLE I.

### B. Security Requirements

The main security threats in EMRs sharing are as follows:
- The disclosure of the identity privacy information: An adversary in the system corrupts several nodes to reveal the real identities of patients.
- Semi-trusted entity: CS, as a semi-trusted entity, is honest but curious. It works according to the protocol, but is inquisitive about the stored healthcare data.
- Collusion attacks: CS conspires with data requestors to obtain certain unauthorized EMRs.

In order to protect the identity privacy of patients while efficiently circulating healthcare data between patients and legal requestors, the EMRs sharing scheme needs to meet the following requirements:
- Efficient sharing: The scheme should provide high efficiency in healthcare data sharing.
- Solving the storage issue of blockchain: For massive medical data, the scheme should be able to break through the storage capacity bottleneck of blockchain.
- Identity privacy preservation: The scheme should be able to protect the identity privacy of users.
- Conditional identity tracking: In special cases, the scheme should be able to provide a trigger mechanism to conditionally track the real identity of any malicious user.
- Resisting collusion attacks: The scheme should be able to resist collusion between CS and data requestors.

### C. Preliminaries

*1) Shamir Secret Sharing Scheme:* Shamir secret sharing scheme is also known as threshold secret sharing scheme, where $N$ is the number of the participants who master secret shares and $t$ is the threshold for secret recovery. In the scheme,

TABLE I:
Symbols and Descriptions

| Symbols | Descriptions | Symbols | Descriptions |
|---|---|---|---|
| $\lambda$ | Security parameter | $msk$ | The private key of $MN$ |
| $q$ | A big prime number | $PK_M$ | The public key of $MN$ |
| $G_1$ | A multiplicative cyclic group of order $q$ | $url$ | Download address |
| $G_2$ | A multiplicative cyclic group of order $q$ | $pk_C$ | The public key of $CS$ |
| $H_1, H_2, H_3, H_4, H_5$ | Hash functions | $sk_c$ | The private key of $CS$ |
| $g, h$ | The generators of $G_1$ | $CS$ | Cloud server |
| $CN_i$ | Consensus node | $t$ | Threshold |
| $x_i$ | The random number of $CN_i$ | $r_{A \rightarrow B}$ | Re-encryption key |
| $sk_i$ | The private key of $CN_i$ | $PID_A$ | The pseudo identity of $Alice$ |
| $pk_i$ | The public key of $CN_i$ | $EMR_A$ | The electronic medical record of $Alice$ |
| $MN$ | Management node | $share_i$ | The secret share of $CN_i$ |

a secret $s$ is divided into $N$ shares, each share is encrypted and sent to each participant. Only when no less than $t$ participants contribute their secret shares, the secret $s$ can be recovered by Lagrange interpolation polynomial. The specific process of the scheme is as follows:

- Setup: $P = \{P_1, \cdots, P_i, \cdots, P_N\}$ is the set of participants, and $q$ is a large prime number. The data owner $D$ selects a secret $s$ and $t-1$ random elements $a_1, a_2, \cdots, a_{t-1}$ from finite field $F_q$, where $a_{t-1} \neq 0$.
- Share distribution: The data owner $D$ uses the above parameters to construct a $t-1$-degree polynomial $f(x) = a_{t-1}x^{t-1} + a_{t-2}x^{t-2} + \cdots + s$. Then, s/he calculates $y_i = f(x_i)$ and individually sends it to the corresponding participant in secure channel, where $x_i$ is the serial number of $P_i$.
- Secret reconstruction: When $t$ or more participants provide their shares, the original polynomial can be reconstructed by Lagrange interpolation polynomial and the secret value can be derived by $s = f(0)$.

$$f(x) = \sum_{i=1}^{t} y_i \prod_{j=1, j \neq i}^{t} \frac{x - x_j}{x_i - x_j} \quad (1)$$

*2) Proxy Re-encryption:* Proxy re-encryption based on public key cryptography allows a third-party agent to convert the ciphertexts encrypted by data owners into the ciphertexts that can be decrypted by data requestors. The agent can only convert the encrypted data in ciphertext state without obtaining plaintext information. Proxy re-encryption can provide accurate access control of EMRs in data outsourcing. The formal definition of proxy re-encryption is as follows:

- $KeyGen(1^k) \rightarrow (PK_A, sk_A, PK_B, sk_B)$: Key generation algorithm is performed by a Key Generation Center (KGC). KGC generates the public and private keys for data owners and data requestors using the security parameter $k$ as input.
- $Encryption(M, PK_A) \rightarrow CT_A$: Encryption algorithm is performed by data owners. It takes the plaintext and public key of a data owner as input and outputs a ciphertext $CT_A$. $CT_A$ is stored by a third-party agent such as CS.
- $ReKeyGen(sk_A, PK_B) \rightarrow rk_{A \rightarrow B}$: Re-encryption key generation algorithm is performed by data owners. It takes the private key of the data owner and the public key of a specific data requestor as input to generate a re-encryption key $rk_{A \rightarrow B}$ for the data requestor. Then, the re-encryption key is sent to the third-party agent.
- $ReEncryption(CT_A, rk_{A \rightarrow B}) \rightarrow CT_B$: Re-encryption algorithm is performed by the third-party agent. It uses the ciphertext $CT_A$ and the re-encryption key $rk_{A \rightarrow B}$ as input to generate a converted ciphertext $CT_B$.
- $Decryption(CT_B, sk_B) \rightarrow M$: Decryption algorithm is performed by the data requestor who can decrypt the converted ciphertext $CT_B$ with his/her own private key.

## IV. THE PROPOSED SCHEME

### A. System Initialization

Given a security parameter $\lambda$ and a large prime $q$, $G_1$ and $G_2$ are two $q$-order multiplicative cyclic groups. $g$ and $h$ are two generators of $G_1$. $e : G_1 \times G_1 \rightarrow G_2$ is a bilinear pairing. $H_1 : G_1 \rightarrow \{0,1\}^{4\lambda}$, $H_2 : \{0,1\}^* \rightarrow Z_q^*$, $H_3 : G_1 \rightarrow G_1$, $H_4 : G_2 \times G_1 \rightarrow G_2$, $H_5 : G_2 \times G_1 \rightarrow Z_q^*$ are five collision-resistant hash functions. MN selects the private key $msk \in Z_q^*$ at random and computes the corresponding public key $PK_M = g^{msk}$. The set of consensus nodes is $\{CN_1, CN_2, ..., CN_N\}$, and the threshold is $t$. $CN_m$ is selected as the $Leader$, and the remaining CNs are $Followers$, where $m = (Num \mod N) + 1$, $Num$ is the height of the current block. Whenever a $Leader$ is elected in CNs, it needs to choose a random number $g_m \in G_1$ and broadcast it publicly. That is to say, $g_m$ is a public parameter generated periodically. $CN_i$ obtains the private key $sk_i$ by randomly selecting $x_i \in Z_q^*$ and calculates the public key $pk_i = g^{sk_i}$. CS randomly selects $x_c \in Z_q^*$ as its private key $sk_c$ and calculates the corresponding public key $pk_C = g^{sk_c}$. The public system parameters are

$\{\lambda, q, G_1, G_2, g, h, e, H_1, H_2, H_3, H_4, H_5, PK_M, \langle pk_i\rangle_{i=1\sim N},$
$pk_c\}.$

## B. User Registration

Suppose Alice is registering as a legitimate UN.

- Registration information generation: Firstly, Alice randomly selects $x_A \in Z_q^*$ as her private key $sk_A$ and calculates the corresponding public key $pk_A = g^{sk_A}$. Meanwhile, She needs to organize her real identity information {*Name*, *ID*, *E-mail*, *Phone*} and constructs $Info_A = I_1||I_2||I_3||I_4$. Here, $I_1, I_2, I_3, I_4$ are the binary representations of four identity information fields with the length of $\lambda$ bits. In other words, each identity information field is limited to $\lambda/16$ characters. If the length is insufficient, the high position should be padded 0. Then, Alice randomly selects $s \in Z_q^*$ and computes $S = g^s$. Next, she calculates protection information $\pi_A = H_1(S) \oplus Info_A$ of the real identity. To share the secret $s$, Alice chooses $t-1$ random numbers $a_1, a_2, \cdots, a_{t-1}$ and sets $a_0 = s$. Subsequently, she constructs a t-degree polynomial $f(x) = \sum_{i=0}^{t-1} a_i x^i$ and calculates polynomial shares $\{f(1), f(2), \cdots, f(N)\}$ for CNs. Finally, she encrypts the corresponding share of each CN.

$$Y_i = pk_i^{f(i)} (1 \le i \le n) \qquad (2)$$

To provide relevant validation information, Alice calculates the commitments $\langle C_j\rangle_{j=0\sim t-1}$ of all polynomial parameters, the commitments $\langle X_i\rangle_{i=1\sim N}$ of all shares, and the authentication information $\langle R_i\rangle_{i=1\sim N}$.

$$\begin{cases} C_j = h^{a_j} (0 \le j < t) \\ X_i = h^{f(i)} (1 \le i \le n) \\ R_i = e(X_i, pk_i) (1 \le i \le N) \end{cases} \qquad (3)$$

Alice broadcasts $\{\pi_A, \langle C_j\rangle_{j=0\sim t-1}, \langle X_i\rangle_{i=1\sim N},$ $\langle R_i\rangle_{i=1\sim N}, \langle Y_i\rangle_{i=1\sim N}\}$ to the entire MCB network.

- Verification: Using public information, each CN can not only verify the correctness of its own received share but also check whether all shares are consistent, so as to ensure that the data owner is honest in the process of share distribution. In order to improve the efficiency of verification, we construct a public batch verification method based on bilinear pairing. The detailed steps are as follows:

  Firstly, in order to check if the share in the commitment $X_i$ is generated according to the polynomial constructed above, $CN_i$ calculates $X_i^*$ to determine whether it is equal to the public commitment $X_i$.

$$X_i^* = \prod_{k=0}^{t-1} (C_k)^{i^k} \qquad (4)$$

Meanwhile, each CN uses public $\langle R_i\rangle_{i=1\sim N}$ and $\langle Y_i\rangle_{i=1\sim N}$ to conduct batch verification of all shares.

$$\prod_{i=1}^{N} R_i = e(h, \prod_{i=1}^{N} Y_i) \qquad (5)$$

If the above formula holds, $CN_i$ recognizes that all shares among CNs are correct. After verifying all shares, $CN_i$ sends a confirmation message with $\alpha_i = H_2(Y_i||sk_i)$ to MN.

- Pseudo identity generation: After receiving all confirmation messages from CNs, MN computes $\alpha = \alpha_1 + \alpha_2 + ... + \alpha_N$. Then, it selects a random number $\beta \in Z_q^*$ and calculates the pseudo identity $PID_A$ of Alice, hash value $\delta$ and signature $\sigma$ by the following formulas:

$$\begin{cases} PID_A = g^{\alpha(\beta + H_2(\pi_A))} \\ \delta = H_3(PID_A) \\ \sigma = \alpha(\beta + H_2(\pi_A)) + \delta msk \end{cases} \qquad (6)$$

Finally, MN sends $\{PID_A, \sigma, \delta\}$ to Alice.

Upon receiving the message from MN, Alice calculates $\delta^* = H_3(PID_A)$ and verifies whether $\delta^* = \delta$ and $g^\sigma = (PK_M)^{\delta^*} \cdot PID_A$ hold. If they hold, Alice keeps $PID_A$ as her own pseudo identity.

- Tracking information recording: MN initiates an upchain request of tracking information $\{\pi_A, PID_A\}$ on the blockchain. $CN_i$ additionally stores the corresponding ciphertext $Y_i$ distributed by Alice. If Alice has malicious behaviors, CNs can retrieve her corresponding identity protection information $\pi_A$ on the blockchain according to $PID_A$. Once the number of CNs that consider Alice as a malicious user exceeds the threshold $t$, the real identity of Alice can be revealed.

## C. Encryption of Electronic Medical Records

After receiving $EMR_A$ of diagnosis, Alice selects a random number $r_A \in Z_q^*$ and obtains the public random number $g_m$ from the current *Leader*. Then, she encrypts $EMR_A$ by the following formulas:

$$\begin{cases} R_1 = e(g, (g_m)^{r_A}) = e(g, g_m)^{r_A} \\ R_2 = H_4(R_1, pk_A) \\ C = R_2 \cdot EMR_A \end{cases} \qquad (7)$$

Alice keeps $R_1$ and $R_2$ securely and calculates the signature $s_A = e(g^{\delta_A}, (g_m)^{x_A})$ with hash value $\delta_A = H_5(C, PID_A)$. Finally, Alice sends the message $\{C, \delta_A, s_A\}$ to CS.

## D. Storage of Electronic Medical Records

Upon receiving $\{C, \delta_A, s_A\}$ from Alice, CS calculates $\delta_A^* = H_5(C, PID_A)$ firstly and verifies if $\delta_A^* = \delta_A$. Then, CS checks the signature by the following formula to determine whether the message comes from Alice.

$$e(pk_A, (g_m)^{\delta_A^*}) = s_A \qquad (8)$$

If the verification passes, CS stores $\{C, \delta_A, s_A\}$ and generates a download address $url$ of $EMR_A$ for Alice.

## E. Record of Metadata on Blockchain

To alleviate the storage pressure of blockchain, metadata only includes the download address $url$, the summary $Info_{Abstract}$ of $EMR_A$ and the pseudo identity $PID_A$ of Alice. Alice initiates an up-chain request for recording her metadata on the blockchain. The detailed process is as follows:

- Alice generates the metadata $M = \{url, Info_{Abstract}, PID_A\}$ and broadcasts the up-chain request $Tx\_Request = \{M, h_A, Sig_A(h_A)\}$ to the whole MCB network, where $h_A$ is the hash value of $M$, $Sig_A(h_A)$ is Alice's signature on $h_A$.
- On receiving $Tx\_Request$, CNs verify the integrity of $M$ and $Sig_A(h_A)$ independently. If it is valid, each CN will send a confirmation $Tx\_Ack$ with the result of verification to $Leader$.
- $Leader$ checks all received $Tx\_Ack$. If the number of $Tx\_Ack$ with "true" is not less than the threshold $t$, $Leader$ puts the metadata $M$ into the transaction pool as an entry.
- When the current consensus interval ends, $Leader$ collects all entries in the transaction pool and packages them. Then, $Leader$ constructs a block request message $Block\_Request = \{block_{Num}, g_m, h_L, Sig_L(h_L)\}$, where $Num$ is the the height of blocks, $g_m$ is the random number published by the current $Leader$, $h_L$ is the hash value of $M$, $Sig_L(h_L)$ is the signature of the current $Leader$. Finally, $Leader$ broadcasts $Block\_Request$ to the whole MCB network.
- Upon receiving $Block\_Request$, each $Follower$ independently verifies the data format, Merkle root of transactions, the integrity of block data, the validity of signature and checks if $g_m$ is the same as that published by the current $Leader$. If all verifications pass, each $Follower$ sends a confirmation $Block\_Ack$ with the result of verification to $Leader$.
- $Leader$ checks all received $Block\_Ack$. If the amount of $Block\_Ack$ with "true" exceeds $t$, $Leader$ notifies all nodes that it uploads the block with height $Num$ to the blockchain.
- After the block is uploaded successfully, a new $Leader$ is determined by the formula $m = (Num \bmod N) + 1$.

## F. Application, Authorization and Access

Suppose that Bob wants to access $EMR_A$, he needs Alice's authorization.

- Application: Bob retrieves the relevant information of $EMR_A$ on the blockchain. Then, he calculates $(g_m)^{x_B}$ and sends an application message $\{PID_B, (g_m)^{x_B}, h_B, Sig_B(h_B)\}$ to Alice, where $g_m$ is the random number recorded on the block associated with $EMR_A$, $x_B$ is Bob's private key, $h_B$ is the hash of $PID_B$ and $(g_m)^{x_B}$, $Sig_B(h_B)$ is Bob's signature on $h_B$.
- Authorization: On receiving the application message of Bob, if Alice allows him to access $EMR_A$, she calculates the re-encryption key $rk_{A \to B}$ by the following formula:

$$rk_{A \to B} = ((g_m)^{x_B})^{r_A/x_A} \qquad (9)$$

Alice encrypts the re-encryption key with CS's public key $pk_C$ and transmits it to CS.
- Re-encryption: CS first recovers the re-encryption key $rk_{A \to B}$ with its private key. Then, CS computes $R_{A \to B}$ and combines it with the ciphertext $C$ of $EMR_A$.

$$R_{A \to B} = e(pk_A, rk_{A \to B}) = e(g, g_m)^{r_A x_B} \qquad (10)$$

- Decryption: Bob downloads the combination of $\{C, R_{A \to B}\}$ and hash value $\delta_A$ via the corresponding $url$. Then, he calculates $\delta_A^* = H_5(C, PID_A)$ and compares it with $\delta_A$ to verify the integrity of $EMR_A$. If the verification succeeds, Bob can finally decrypt $C$ to access $EMR_A$ by the following formulas:

$$\begin{cases} R_1^* = (R_{A \to B})^{1/x_B} = e(g, g_m)^{r_A} \\ R_2^* = H_4(R_1^*, pk_A) \\ EMR_A = C/R_2^* \end{cases} \qquad (11)$$

## G. Tracking of Malicious Nodes

Smart contract is an essential component of blockchain which is an event-driven program deployed on blockchain. Once meeting the trigger conditions, it will automatically execute the predefined process in the contract.

In our scheme, smart contract is used to automatically track malicious nodes. When a node in the system has malicious behaviors, CNs that identify the malicious behaviors will form a set which is called the authorized subset. When the number of CNs in the authorized subset exceeds the threshold $t$, the tracking process will be initiated. $CN_i$ in the authorized subset firstly uses its own private key $sk_i$ to recover $share_i$ from the ciphertext $Y_i$.

$$share_i = Y_i^{1/sk_i} = g^{f(i)} \qquad (12)$$

In order to ensure that the provided $share_i$ is indeed from the corresponding $Y_i$, $CN_i$ also needs to put forward relevant proof information. Firstly, $CN_i$ chooses a random number $r_i \in Z_q^*$ and calculates $B_1^{(i)} = (share_i)^{r_i}$, $B_2^{(i)} = (g)^{r_i}$. Then, $CN_i$ calculates challenge $c_B^{(i)} = H_2(share_i||g||Y_i||pk_i||B_1^{(i)}||B_2^{(i)})$ and $b_B^{(i)} = r_i + c_B^{(i)} sk_i$.

Each CN in the authorized subset submits its own $\{c_B^{(i)}, b_B^{(i)}, share_i, \pi_A\}$ to the smart contract. Algorithm 1 gives the detailed steps.

When more than $t$ members in the authorized subset input their shares into the smart contract, the tracking process will be automatically triggered. Finally, Alice's real identity $Info_A$ can be revealed by the following formula:

$$Info_A = \pi_A \oplus H_1(S) \qquad (13)$$

## V. Security Analysis

### A. Identity Privacy Preservation and Traceability

*Lemma 1:* Under the DL assumption and CDH assumption, it is hard to obtain $share_i$ through the public commitment $X_i$ and the ciphertext $Y_i$ disclosed in the registration process.

---

**Algorithm 1** Tracking of Malicious Nodes

**Input:** $\pi_A$, $[share_1, share_2, \ldots, share_t]$,
　　$[c_B{}^{(1)}, c_B{}^{(2)}, \ldots, c_B{}^{(t)}]$, $[b_B{}^{(1)}, b_B{}^{(2)}, \ldots, b_B{}^{(t)}]$
**Output:** $\pi_A \oplus H_1(S)$

1: **for** i=1 to t **do**
2: 　　Compute:
$$c_B{}^{(i)*} = H_2(share_i||g||Y_i||pk_i||(share_i)^{b_B{}^{(i)}}(Y_i)^{-c_B{}^{(i)}}$$
$$||g^{b_B{}^{(i)}}(Y_i)^{-c_B{}^{(i)}})$$
3: 　　**if** $c_B{}^{(i)*} \neq c_B{}^{(i)}$ **then**
4: 　　　　fail
5: 　　**end if**
6: **end for**
7: long $S = 1$
8: **for** i=1 to t **do**
9: 　　long $L_i = 1$
10: 　　**for** $j \in [1, t]$ **do**
11: 　　　　**if** $j \neq i$ **then**
12: 　　　　　　$L_i = L_i \cdot \frac{j}{j-i}$
13: 　　　　**end if**
14: 　　**end for**
15: 　　$S = S \cdot pow(S_i, L_i)$
16: **end for**

---

*Proof:* In the registration stage, to be a legal user, Alice needs to broadcast $\langle X_i \rangle_{i=1 \sim N}$ and $\langle Y_i \rangle_{i=1 \sim N}$ to the whole MCB network. That is, $X_i$ and $Y_i$ are also available to any adversary. Now, we assume that a adversary tries to make use of these public information $\{g, h, X_i, Y_i, pk_i\}$ to reveal $share_i$.

To simplify the proof, let $g = h^\alpha$, $X_i = h^{f(i)} = h^\beta$ and $pk_i = g^{sk_i} = h^{\alpha sk_i} = h^\gamma$, so that we can obtain the ciphertext $Y_i = pk_i^{f(i)} = h^{\beta\gamma}$. The adversary tries to recover $share_i = g^{f(i)} = h^{\alpha f(i)} = h^{\alpha\beta}$. Therefore, the problem is transformed to calculate $h^{\alpha\beta}$, given $h^\beta$, $h^\gamma$ and $h^{\beta\gamma}$ for any $\alpha, \beta, \gamma \in Z_q^*$. With all public information in hand, the adversary can calculate $h^{\alpha\beta}$ in two ways:

- Using $h^\alpha$ and $h^\beta$: Suppose the adversary can calculate $h^{\alpha\beta}$ directly from $h^\alpha$ and $h^\beta$. According to the CDH assumption, given $h$, $h^\alpha$ and $h^\beta$ for any $\alpha, \beta \in Z_q^*$, there exists no probabilistic polynomial time adversary to calculate $h^{\alpha\beta}$ by a non-negligible advantage. This way contradicts the CDH assumption.
- Using $h^\alpha$, $h^\gamma$ and $h^{\beta\gamma}$: Suppose the adversary can also use $h^\gamma$ and $h^{\beta\gamma}$ to get $\beta$, and then use $(h^\alpha)^\beta$ to obtain $h^{\alpha\beta}$. According to the DL assumption, given $h^\gamma$ and $h^{\beta\gamma}$ for any $\gamma \in Z_q^*$, there exists no probabilistic polynomial time adversary to calculate $\beta$ by a non-negligible advantage. If the adversary cannot obtain $\beta$, s/he cannot further get $h^{\alpha\beta}$.

For above, it is proved that, relying on the DL assumption and CDH assumption, the adversary cannot obtain $share_i$ through the commitment $X_i$ and the ciphertext $Y_i$.

*Lemma 2:* Under the DL assumption and CDH assumption, any adversary cannot extract a user's secret even if it corrupts $t-1$ or fewer CNs.

*Proof:* Due to the threshold $t$, our scheme can resist the collusion of at most $t-1$ CNs. In other words, it can resist an adversary who is able to corrupt only $t-1$ participants or less. For generality, we set the serial number of the corrupted CNs as $i = 1 \sim t-1$. In this case, the information obtained by the adversary is: the commitments $\langle X_i \rangle_{i=1 \sim N}$ and the ciphertexts $\langle Y_i \rangle_{i=1 \sim N}$ of shares, the public keys $\langle pk_i \rangle_{i=1 \sim N}$ of CNs, the commitments $\langle C_i \rangle_{i=0 \sim t-1}$ of polynomial parameters, the private keys $\langle sk_i \rangle_{i=1 \sim t-1}$ and shares $\langle share_i \rangle_{i=1 \sim t-1}$ of the corrupted CNs. The ultimate goal of the adversary is to obtain $s$ or $S = g^s$.

Let $g = h^\alpha$ and $C_0 = h^{a_0} = h^s = h^\beta$, then the goal of the adversary is to calculate $\beta$ or $h^{\alpha\beta}$. It can try the following three ways:

- Using $C_0 = h^\beta$: Calculating $\beta$ directly from $h^\beta$ is equivalent to solving the DL hard problem, which contradicts the DL assumption.
- Using $g = h^\alpha$ and $C_0 = h^\beta$: Calculating $h^{\alpha\beta}$ directly from $h^\alpha$ and $h^\beta$ is equivalent to solving the CDH hard problem, which contradicts the CDH assumption.
- Using $\langle share_i \rangle_{i=1 \sim t-1}$: When $t$ or more shares are collected, the adversary can obtain $h^{\alpha\beta}$ by the following formulas:

$$\begin{cases} \prod_{i=1}^{t}(share_i)^{L_i} = \prod_{i=1}^{t}\left(g^{f(i)}\right)^{L_i} = g^{\sum_{i=1}^{t} f(i)L_i} \\ \qquad\qquad = g^{f(0)} = g^s = h^{\alpha\beta} \qquad (14) \\ L_i = \prod_{j=1, j \neq i}^{t} \frac{j}{j-i} \end{cases}$$

According to the Lagrange interpolation theorem, $S$ cannot be reconstructed from the shares of $t-1$ corrupted CNs.

In order to achieve the purpose of restoring $S$, the adversary and the corrupted CNs can also use the information that they have mastered to crack any of shares $\langle share_k \rangle_{k=t \sim N}$ to meet the threshold $t$. From Lemma 1, it shows that, under the DL assumption and CDH assumption, the adversary cannot obtain the share of an uncorrupted CN through the public information. In summary, under the DL assumption and CDH assumption, even if the adversary corrupts $t-1$ or less CNs, s/he cannot restore $S$.

*Lemma 3:* Any adversary cannot reveal the real identity of a legal user through the public identity protection information.

*Proof:* When Alice registers into the system, she uses the constructed secret $S$ to cover up her real identity $Info_A$ by the following formula:

$$\pi_A = H_1(S) \oplus Info_A \qquad (15)$$

According to Lemma 1 and Lemma 2, the constructed secret $S$ cannot be recovered by any adversary. Meanwhile, as long as the security parameter $\lambda$ is strong enough, cracking the real identity information $Info_A$ from the protection information $\pi_A$ is equivalent to violently guessing a string with the strength of $4\lambda$ bits. Therefore, the real identities of users cannot be obtained from the protection information.

*Theorem 1:* Under the DL assumption and CDH assumption, the anonymous and conditionally traceable identity privacy preserving mechanism in the distributed scenario is secure.

*Proof:* In the anonymous and conditionally traceable identity privacy protection mechanism, there are three ways for an adversary to extract the real identities of users:

- The adversary acquires $share_i$ through the public commitment $X_i$ and the ciphertext $Y_i$ during the registration process, and then further obtains the shares $\langle share_i \rangle_{i=1 \sim t}$ that meet the threshold $t$. Next, it recovers the secret $S$ and calculates the real identity $Info_A$ through the public protection information $\pi_A$.
- The adversary has the ability to corrupt at most $t-1$ CNs. Then, s/he uses the private keys and shares of the corrupted CNs with other public information to recover the secret $S$ and the real identity $Info_A$.
- The adversary directly uses the public identity protection information $\pi_A$ to crack the hidden real identity $Info_A$.

$$\prod_{i=1}^{t} (share_i)^{L_i} = S \tag{16}$$

$$Info_A = \pi_A \oplus H_1(S) \tag{17}$$

According to Lemma 1, under the DL assumption and CDH assumption, the adversary cannot obtain share $share_i$ through the commitment $X_i$ and the ciphertext $Y_i$, so that it cannot collect $t$ shares to recover the secret $S$ and calculate the real identity $Info_A$.

According to Lemma 2, under the DL assumption and CDH assumption, even if the adversary corrupts $t-1$ CNs, $S$ cannot be forcibly reconstructed with the guarantee of Lagrange interpolation theorem.

According to Lemma 3, in the case of reasonable security strength, the adversary is impossible to reveal the real identity $Info_A$ from the public protection information $\pi_A$.

In summary, the anonymous and conditionally traceable identity privacy preserving mechanism is proved to be secure. Only when $t$ or more CNs recognize a user's malicious behavior at the same time, the user's identity can be revealed.

## B. Data Confidentiality

In order to reduce the storage pressure of blockchain, the main body of EMRs is stored on a semi-trusted CS. To guarantee the confidentiality of EMRs, the proposed scheme introduces an improved proxy re-encryption algorithm as the core component of EMRs sharing system, in which EMRs are encrypted with data owners' public keys and stored into CS. If a data owner authorizes a data requestor access to his/her EMRs, s/he needs to generate a re-encryption key for the requestor and send it to CS. Then, CS uses the re-encryption key to convert the ciphertexts of EMRs, so the data requestor can decrypt through its own private key. During the whole process, CS only deals with the ciphertexts of EMRs and the re-encryption key. It cannot obtain any useful information through the original ciphertexts and the re-encrypted ciphertexts. Therefore, the proposed scheme achieves the confidentiality of the outsourced data.

## C. Anti-collusion Security

The proposed scheme can effectively resist collusion attacks between a semi-trusted CS and any data requestor who has access to certain EMRs. In the scheme, CS is responsible for storing the ciphertexts of EMRs and converting the relevant ciphertexts for data requestors. As a semi-trusted entity, although CS will perform its duties step by step, it may spoon on the stored data, conspiring with data requestors via re-encryption keys to grab the unauthorized EMRs.

In the proposed scheme, the consensus mechanism ensures that $Leader$ is selected in CNs according to the formula $m = (Num \bmod N) + 1$. Then, $Leader$ exposes a random number $g_m$ for users to encrypt EMRs. Without loss of generality, we assume that the first $Leader1$ selects a random number $g_{m1}$ and records it in the block with height $Num_1$. Meanwhile, the metadata of EMRs encrypted with $g_{m1}$ is also recorded in this block. If a data requestor wants to retrieve an EMR associated with the block, s/he needs to use $g_{m1}$ to construct the application information. When the current consensus interval ends, the second $Leader2$ will be selected in turn. It publishes a new random number $g_{m2}$ which is recorded in the block with height $Num_2$. The metadata of EMRs encrypted with $g_{m2}$ will also be recorded in this block. Here, Alice may have multiple EMRs generated in different periods, so the metadata of these EMRs are recorded in the blocks with different heights. Suppose that Bob wants to access Alice's $EMR_1$, in which the metadata of $EMR_1$ exists in the block with height $Num_1$, he needs to use $g_{m1}$ and his private key $x_B$ to calculate $(g_{m1})^{x_B}$. Then, Bob initiates an access request with $(g_{m1})^{x_B}$ to Alice. If Alice allows Bob to access $EMR_1$, she uses $(g_{m1})^{x_B}$ and her own private key to generate the re-encryption key $rk_{A \to B} = ((g_{m1})^{x_B})^{r_A/x_A}$ and sends it to CS. On receiving the re-encryption key, CS converts the ciphertext of $EMR_1$. Finally, Bob downloads the ciphertext through $url$ and decrypt it by his own private key.

Unfortunately, many existing schemes do not achieve anti-collusion security. If the data owner authorizes a data requestor to access any of EMRs, the data requestor might conspire with CS to illegally obtain other unauthorized EMRs. In the proposed scheme, the EMRs of Alice are recorded in the blocks with different heights, so they are encrypted with different random numbers. Suppose that the metadata of $EMR_2$ is recorded in the block with height $Num_2$, and the random number used to protect $EMR_2$ is $g_{m2}$. In the encryption stage, $EMR_2$ is encrypted according to the following formulas:

$$\begin{cases} R_1^{(2)} = e(g, (g_{m2})^{r_A}) \\ R_2^{(2)} = H_4(R_1^{(2)}, pk_A) \\ C^{(2)} = R_2^{(2)} \cdot EMR_2 \end{cases} \tag{18}$$

Without the authorization of $EMR_2$, Bob can only obtain the information $(g_{m1})^{r_A/x_A}$ by collusion. If Bob attempts to access the unauthorized $EMR_2$, he can calculate $e(pk_A, (g_{m1})^{r_A/x_A}) = e(g, g_{m1})^{r_A}$. Because it is different from $R_1^{(2)} = e(g, g_{m2})^{r_A}$, Bob cannot further calculate $R_2^{(2)}$ from $e(g, g_{m1})^{r_A}$. So, the information $(g_{m1})^{r_A/x_A}$ that Bob acquires by colluding with CS is not valid for other EMRs.

Therefore, the proposed scheme can resist collusion attacks between a semi-trusted CS and data requestors.

### D. Correctness Analysis

*Lemma 4:* Any verifier can confirm shares are generated from the polynomial $f(x)$ by checking the formula $X_i^* = \prod_{k=0}^{t-1}(C_k)^{i^k} = X_i$.

*Proof:* In the registration process, each UN needs to publish the commitments $\langle C_j \rangle_{j=0 \sim t-1}$ of polynomial parameters $a_0, a_1, a_2, ..., a_{t-1}$ and the commitments $\langle X_i \rangle_{i=1 \sim N}$ of all shares. Any verifier can calculate and compare $X_i^*$ with the commitment $X_i$ to verify whether the $f(i)$ hidden in $X_i$ is calculated via the polynomial $f(x)$.

$$X_i^* = \prod_{k=0}^{t-1}(C_k)^{i^k} = \prod_{k=0}^{t-1}(h)^{a_k i^k} = h^{\sum_{k=0}^{t-1} a_k i^k} = h^{f(i)} = X_i$$
$$(19)$$

*Lemma 5:* Any verifier can verify the authenticity of all shares in batches through the authentication information $\langle R_i \rangle_{i=1 \sim N}$ and the ciphertexts $\langle Y_i \rangle_{i=1 \sim N}$ of all shares.

*Proof:* Verifiers check if $f(i)$ in the commitment $X_i = h^{f(i)}$ is consistent with $f(i)$ in the ciphertext $Y_i = pk_i^{f(i)}$ to ensure that CNs do provide correct shares when reconstructing the secret.

Since CNs must independently verify the authenticity of all shares, to avoid redundant calculations of CNs, we enable the data owner to pre-calculate the authentication information $\langle R_i \rangle_{i=1 \sim N}$ and broadcast them.

Each CN firstly calculates $\prod_{i=1}^{N} R_i$.

$$\prod_{i=1}^{N} R_i = \prod_{i=1}^{N} e(X_i, pk_i) = \prod_{i=1}^{N} e(h^{f(i)}, g^{sk_i})$$
$$= \prod_{i=1}^{N} e(h, g)^{f(i)sk_i} = e(h, g)^{\sum_{i=1}^{N} f(i)sk_i}$$
$$(20)$$

Then, they use the public ciphertexts $\langle Y_i \rangle_{i=1 \sim N}$ of all shares to calculate $e(h, \prod_{i=1}^{N} Y_i)$, respectively.

$$e(h, \prod_{i=1}^{N} Y_i) = e(h, Y_1 \cdot Y_2 \cdots Y_N)$$
$$= e(h, Y_1)e(h, Y_2) \cdots e(h, Y_N)$$
$$= e(h, g^{f(1)sk_1})e(h, g^{f(2)sk_2}) \cdots e(h, g^{f(N)sk_N})$$
$$= e(h, g)^{f(1)sk_1 + f(2)sk_2 + \cdots + f(N)sk_N}$$
$$= e(h, g)^{\sum_{i=1}^{N} f(i)sk_i}$$
$$(21)$$

From the derivation of the above two formulas, $\prod_{i=1}^{N} R_i$ is ultimately equal to $e(h, \prod_{i=1}^{N} Y_i)$. Therefore, verifiers can use the authentication information $\langle R_i \rangle_{i=1 \sim N}$ and the ciphertexts

$\langle Y_i \rangle_{i=1 \sim N}$ to check the authenticity of all shares in batches through the following formula:

$$\prod_{i=1}^{N} R_i = e(h, \prod_{i=1}^{N} Y_i)$$
$$(22)$$

*Lemma 6:* If $t$ or more correct shares are collected, the constructed secret can be restored correctly.

*Proof:* The secret can be reconstructed by the following formula:

$$\prod_{i=1}^{t}(share_i)^{L_i} = \prod_{i=1}^{t}\left(g^{f(i)}\right)^{L_i}$$
$$= g^{\sum_{i=1}^{t}(f(i) \prod_{j=1, j \neq i}^{t} \frac{j}{j-i})}$$
$$= g^{f(0)} = g^s = S$$
$$(23)$$

### E. Functional Feature

In this part, the proposed scheme is compared with the other five blockchain-based data sharing schemes in terms of security features. These schemes are similar in architecture and all take proxy re-encryption technology as the core component of EMRs sharing. But each scheme has its own unique features to meet different security requirements. We compare the functional features of these schemes in terms of the following indicators.

- Decentralization: It refers that the scheme does not require a centralized trusted entity.
- Confidentiality: It refers that healthcare data should not be disclosed to unauthorized users, even a third semi-trusted party, in the process of storing or sharing.
- Integrity: It refers that the scheme can ensure the integrity of the stored and shared data.
- Identity privacy: It refers that the scheme can ensure the identity privacy of patients in the process of sharing EMRs.
- Identity tracking: It refers that the scheme provides a secure and reliable mechanism to track the identities of malicious users if necessary.
- Non-interactive: It refers that the scheme does not require complex interaction between data owners and data requestors in the process of application and authorization.
- Smart contract: It refers that the scheme can deploy smart contracts securely and flexibly.
- Anti-collusion security: It refers that the scheme can resist collusion attacks launched by semi-trusted third parties and data requestors who conspire to illegally recover unauthorized ciphertexts.

TABLE II shows that our scheme meets all functional features. In particular, our scheme can resist collusion attacks and realize the conditional identity tracking mechanism. Although the scheme in [33] also realizes anti-collusion security, it is too idealistic to suppose that the third-party is completely credible. Under the assumption of semi-trusted entities, our scheme achieves anti-collusion security while the other schemes fail to meet this feature. Moreover, our scheme provides an

TABLE II: Comparison of functional features

| Scheme | Decentralization | Confidentiality | Integrity | Identity privacy | Identity tracking | Non-interactive | Smart contract | Anti-collusion security |
|--------|------------------|-----------------|-----------|------------------|-------------------|-----------------|----------------|-------------------------|
| [26] | √ | √ | √ | √ | × | √ | × | × |
| [29] | √ | √ | √ | × | × | √ | × | × |
| [30] | √ | √ | √ | × | × | √ | × | × |
| [32] | × | √ | √ | × | × | √ | × | × |
| [33] | × | √ | √ | √ | × | × | × | √ |
| Ours | √ | √ | √ | √ | √ | √ | √ | √ |



Fig. 2: Comparison of computational overhead in encryption stage
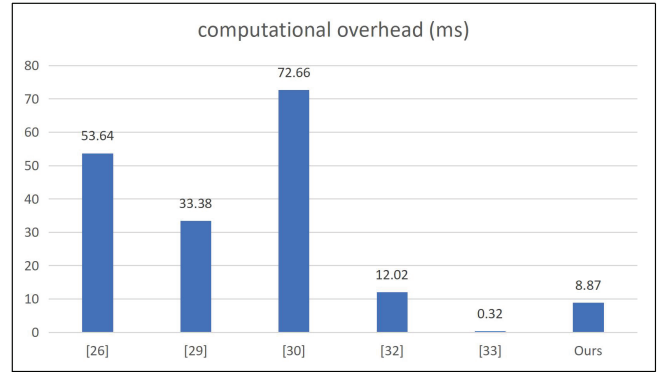


Fig. 3: Comparison of computational overhead in the re-encryption key generation stage

anonymous and conditionally traceable mechanism for privacy preservation that is a crucial security feature in healthcare data sharing scenarios.

## VI. PERFORMANCE ANALYSIS

In this section, the proposed scheme is compared with the selected schemes in terms of computational overhead. For better interaction and circulation of EMRs, the healthcare data management system should have excellent performance. Especially in the case of massive data, an efficient and secure EMRs sharing scheme may greatly promote the quality of medical services. Therefore, the detailed performance simulation and quantitative analysis are given in this section.

Our scheme and the selected schemes all involve proxy re-encryption as the core component. The formal model of proxy re-encryption is mainly divided into five stages: initialization, encryption, re-encryption key generation, re-encryption and decryption. Therefore, we simulate and measure the computational overhead of all schemes in the stages of encryption, re-encryption key generation, re-encryption and decryption. According to the actual application scenarios, we simulate each scheme via the tool of Java Pairing Based Cryptography (JPBC) library. The simulation environment is a PC with Intel(R) Core(TM) I5-8250U CPU rated at 1.80GHz and 8.00GB memory running 64-bit Windows operating system. We test the average time consumption in the stages of encryption, re-encryption key generation, re-encryption and decryption.

According to Fig. 2, in the encryption stage, the performance of our scheme is at the medium level. In the re-encryption key generation stage, Fig. 3 shows that the performance of our scheme is only the second to the scheme in
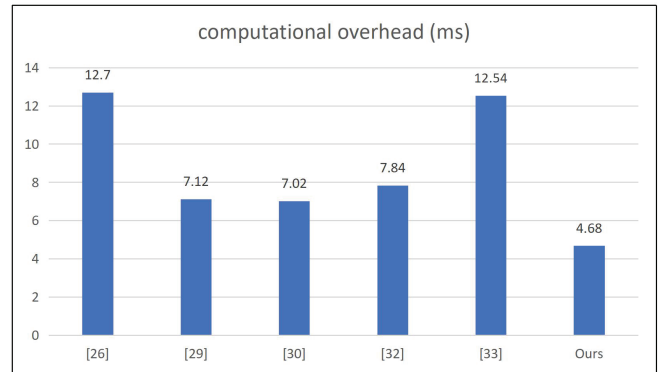


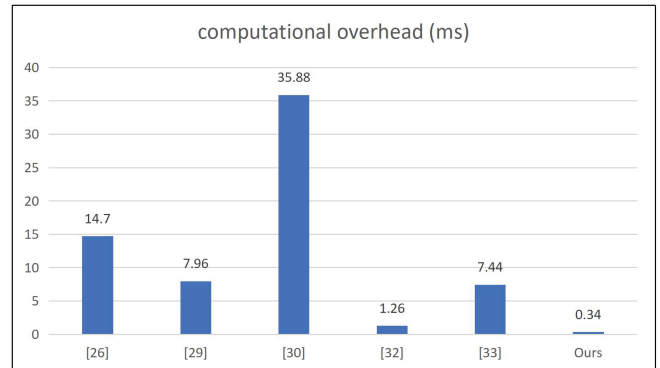Fig. 4: Comparison of computational overhead in the re-encryption stage



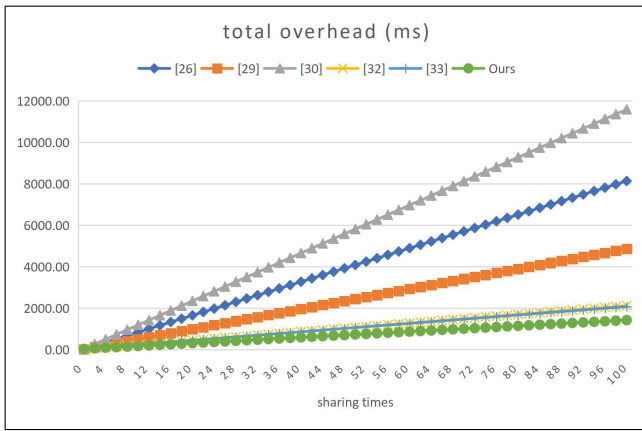Fig. 5: Comparison of computational overhead in decryption stage

Fig. 6: Comparison of total overhead

[33]. But in Fig. 4 and Fig. 5, the computational overhead of our scheme is the lowest in the stages of re-encryption and decryption. From above comparison, the performance of our scheme is excellent in the stages of re-encryption key generation, re-encryption and decryption. Although the performance of our scheme in the encryption stage is at the medium level, all EMRs only need to be encrypted and stored once.

By measuring the computing time consumption of each stage, the overall performance of each scheme is analyzed and evaluated. In the MCB system, users from different places join the system, so the number of users may be quite large. Meanwhile, EMRs are often shared many times with different requestors. Since the initialization only runs once when the system starts up, compared with the multiple sharing of EMRs by a large number of users, the overhead of initialization stage can be ignored. We assume that all schemes have the same number of users and EMRs, the overall performance of each system will change with the times of sharing EMRs. We represent the overhead of encryption, re-encryption key generation, re-encryption and decryption as $cost_{Encryption}$, $cost_{ReKeyGen}$, $cost_{Re-Encryption}$ and $cost_{Decryption}$, respectively. In the healthcare data sharing system, EMRs only need to be encrypted once, but each sharing of EMRs must involve re-encryption key generation, re-encryption and decryption. Therefore, we calculate the total overhead of sharing EMRs n times by the following formula:

$$cost = cost_{Encryption} + n * (cost_{ReKeyGen} + \\ cost_{Re-Encryption} + cost_{Decryption}) \quad (24)$$

As shown in Fig. 6, with the increase of sharing times, the total overhead of our scheme is lower than the other schemes. Therefore, our scheme has higher efficiency in EMRs sharing scenarios.

## VII. CONCLUSION

The healthcare industry is transforming from traditional medicine to digital driven medicine. However, there is still a great challenge to effectively share healthcare data remotely while protecting personal privacy during the medical data

sharing. Therefore, we proposed a secure and efficient remote EMRs sharing scheme over blockchain. The scheme could not only greatly improve the efficiency of remote EMRs sharing but also realize conditional identity privacy preservation and tracking. With the help of the on-chain/off-chain model of blockchain and cloud services, the scheme greatly alleviated the storage pressure of massive medical data. Most importantly, our scheme achieved anti-collusion security that could prevent the semi-trusted cloud server from colluding with data requestors to access unauthorized EMRs. Through simulation and theoretical analysis, our scheme proved to be more efficient than the selected schemes due to the lower computational overhead in EMRs sharing.

## REFERENCES

[1] P. M, T. N. Nguyen, M. Hamdi and K. Cengiz, "A Measurement Approach Using Smart-IoT Based Architecture for Detecting the COVID-19," *Neural Processing Letters*, 2021.

[2] H. Li, K. Yu, B. Liu, C. Feng, Z. Qin and G. Srivastava, "An Efficient Ciphertext-Policy Weighted Attribute-Based Encryption for the Internet of Health Things," *IEEE Journal of Biomedical and Health Informatics (Early Access)*, 2021.

[3] T. N. Nguyen and S. Zeadally, "Mobile Crowd-Sensing Applications: Data Redundancies, Challenges, and Solutions," *ACM Transactions on Internet Technology*, vol. 22, no. 2, 2021.

[4] L. Feng, A. Ali, M. Iqbal, A. K. Bashir, S. A. Hussain and S. Pack, "Optimal haptic communications over nanonetworks for e-health systems," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 5, pp. 3016-3027, 2019.

[5] P. Nagrath, T. N. Nguyen, S. Aggarwal and D. J. Hemanth, "A comprehensive E-commerce customer behavior analysis using convolutional methods," *Computers and Electrical Engineering*, vol. 96, 2021.

[6] T. Greenhalgh, S. Hinder, K. Stramer, T. Bratan and J. Russell, "Adoption, non-adoption, and abandonment of a personal electronic health record: case study of HealthSpace," *British Medical Journal*, vol. 341, no. 7782, pp. 1091-1091, 2010.

[7] B. W. Hesse, D. Hansen, T. Finholt, S. Munson, W. Kellogg and J. C. Thomas, "Social participation in health 2.0," *Computer*, vol. 43, no. 11, pp. 45-52, 2010.

[8] J. Benaloh, M. Chase, E. Horvitz and K. Lauter, "Patient controlled encryption: ensuring privacy of electronic medical records," in Proc. of the 2009 ACM Workshop on Cloud Computing Security, 2009, pp. 103-114.

[9] M. Farzandipour, F. Sadoughi, M. Ahmadi and I. Karimi, "Security requirements and solutions in electronic health records: Lessons learned from a comparative study," *Journal of Medical Systems*, vol. 34, no. 4, pp. 629-642, 2010.

[10] S. Haas, S. Wohlgemuth, I. Echizen, N. Sonehara and G. Mller, "Aspects of privacy for electronic health records," *International Journal of Medical Informatics*, vol. 80, no. 2, pp. 26-31, 2011.

[11] H. Jin, Y. Luo, P. Li and J. Mathew, "A review of secure and privacy-preserving medical data sharing," *IEEE Access*, vol. 7, pp. 61656-61669, 2019.

[12] J. L. Fernndez-Alemn, I. C. Senor, P. A. O. Lozoya and A. Toval, "Security and privacy in electronic health records: A systematic literature review," *Journal of Biomedical Informatics*, vol. 46, no. 3, pp. 541-562, 2013.

[13] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," [online] Available: http://www.bitcoin.org/bitcoin.pdf, 2008.

[14] W. Raghupathi, V. Raghupathi, "Big data analytics in healthcare: Promise and potential," *Health Information Science and Systems*, vol. 2, no. 1, pp. 1-6, 2014.

[15] L. Yang, K. Yu, S. X. Yang, C. Chakraborty, Y. Liu, and T. Guo, "An Intelligent Trust Cloud Management Method for Secure Clustering in 5G enabled Internet of Medical Things," *IEEE Transactions on Industrial Informatics (Early Access)*, 2021.

[16] X. Zhang, S. Postgrad, "Blockchain support for flexible queries with granular access control to electronic medical records (EMR)," in Proc. of 2018 IEEE International Conference on Communications, 2018, pp. 1-6.

[17] S. Cao, G. Zhang, P. Liu, X. Zhang and F. Neri, "Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain," *Information Sciences*, vol. 485, pp. 427-440, 2019.

[18] J. Liu, X. Li, L. Ye, H. Zhang, X. Du and M. Guizani, "BPDS: A blockchain based privacy-preserving data sharing for electronic medical records," in Proc. of 2018 IEEE Global Communications Conference, 2018, pp. 1-6.

[19] Y. Liu, J. Zhang and Q. Gao, "A blockchain-based secure cloud files sharing scheme with fine-grained access control," in Proc. of 2018 International Conference on Networking and Network Applications, 2018, pp. 277-283.

[20] J. Liu, H. Tang, R. Sun, X. Du and M. Guizani, "Lightweight and privacy-preserving medical services access for healthcare cloud," *IEEE Access*, vol. 7, pp. 106951-106961, 2019.

[21] L. Zhu, H. Dong, M. Shen and K. Gai, "An incentive mechanism using shapley value for blockchain-based medical data sharing," in Proc. of 2019 IEEE 5th Intl Conference on Big Data Security on Cloud, 2019, pp. 113-118.

[22] L. Liu, J. Feng, Q. Pei, C. Chen, Y. Ming, B. Shang and M. Dong, "Blockchain-Enabled Secure Data Sharing Scheme in Mobile-Edge Computing: An Asynchronous Advantage ActorCCritic Learning Approach," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2342-2353, 2021.

[23] A. Azaria, A. Ekblaw, T. Vieira and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in Proc. of 2016 2nd International Conference on Open and Big Data, 2016, pp. 25-30.

[24] B. Sharma, R. Halder and J. Singh, "Blockchain-based interoperable healthcare using zero-knowledge proofs and proxy re-encryption," in Proc. of 2020 International Conference on Communication Systems and Networks, 2020, pp. 1-6.

[25] K. Yu, L. Tan, C. Yang, K. K. R. Choo, A. K. Bashir, J. J. P. C. Rodrigues and T. Sato, "A blockchain-based Shamir's threshold cryptography scheme for data protection in industrial internet of things settings," *IEEE Internet of Things Journal (Early Access)*, 2021.

[26] S. Noh, P. Youngho, S. Chur, S. Sang-Uk and R. K. Hyune, "Blockchain-based user-centric records management system," *International Journal of Control and Automation*, vol. 10, no. 11, pp. 133-144, 2017.

[27] C. Sur, C. D. Jung, Y. Park and K. H. Rhee, "Chosen-ciphertext secure certificateless proxy re-encryption," in Proc. of IFIP International Conference on Communications and Multimedia Security, 2010, pp. 214-232.

[28] C. Feng, K. Yu, A. K. Bashir, Y. D. Al-Otaibi, Y. Lu, S. Chen and D. Zhang, "Efficient and secure data sharing for 5G flying drones: A blockchain-enabled approach," *IEEE Network*, vol. 35, no. 1, pp. 130-137, 2021.

[29] Z. Wang, Y. Tian and J. Zhu, "Data sharing and tracing scheme based on blockchain," in Proc. of 2018 8th International Conference on Logistics, Informatics and Service Sciences, 2018, pp. 1-6.

[30] N. Eltayieb, L. Sun, K. Wang and F. Li, "A certificateless proxy re-encryption scheme for cloud-based blockchain," in Proc. of International Conference on Frontiers in Cyber Security, 2019, pp. 293-307.

[31] Y. Sun, J. Liu, K. Yu, M. Alazab and K. Lin, "PMRSS: Privacy-Preserving Medical Record Searching Scheme for Intelligent Diagnosis in IoT Healthcare," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1981-1990, 2022.

[32] T. T. Thwin, S. Vasupongayya, "Blockchain-based access control model to preserve privacy for personal health record systems," *Security and Communication Networks*, vol. 2019, pp. 1-15, 2019.

[33] X. Liu, Z. Wang, C. Jin, F. Li and G. Li, "A blockchain-based medical data sharing and protection scheme," *IEEE Access*, vol. 7, pp. 118943-118953, 2019.

[34] L. Tan, K. Yu, N. Shi, C. Yang, W. Wei and H. Lu, "Towards Secure and Privacy-Preserving Data Sharing for COVID-19 Medical Records: A Blockchain-Empowered Approach," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 1, pp. 271-281, 2022.