Computer Systems Science & Engineering
DOI:10.32604/csse.2021.014557
Article

Tech Science Press

A Secure Signcryption Scheme for Electronic Health Records Sharing in Blockchain

Xizi Peng[1], Jinquan Zhang[1,*], Shibin Zhang[1], Wunan Wan[1], Hao Chen[1] and Jinyue Xia[2]

[1]School of Cybersecurity, Chengdu University of Information Technology, Chengdu, 610225, China
[2]International Business Machines Corporation (IBM), New York, 10041, NY, 212, USA
*Corresponding Author: Jinquan Zhang. Email: zhjqcom@163.com
Received: 28 September 2020; Accepted: 28 November 2020

Abstract: In the existing Electronic Health Records (EHRs), the medical information of patients is completely controlled by various medical institutions. As such, patients have no dominant power over their own EHRs. These personal data are not only inconvenient to access and share, but are also prone to cause privacy disclosure. The blockchain technology provides a new development direction in the medical field. Blockchain-based EHRs are characterized by decentralization, openness and non-tampering of records, which enable patients to better manage their own EHRs. In order to better protect the privacy of patients, only designated receivers can access EHRs, and receivers can authenticate the sharer to ensure that the EHRs are real and effective. In this study, we propose an identity-based signcryption scheme with multiple authorities for multiple receivers, which can resist N-1 collusion attacks among N authorities. In addition, the identity information of receivers is anonymous, so the relationship between them and the sharer is not disclosed. Under the random oracle model, it was proved that our scheme was secure and met the unforgeability and confidentiality requirements of signcryption. Moreover, we evaluated the performance of the scheme and found that it had the moderate signcryption efficiency and excellent signcryption attributes.

Keywords: Electronic health records; blockchain; identity-based signcryption; multiple authorities; multiple receivers

1 Introduction

Electronic Health Records (EHRs) are to digitize the paper-based health records, so that they can be stored, retrieved and accessed more conveniently and quickly in a network. However, some problems in the existing EHRs remain to be solved. Firstly, EHRs of patients are mainly stored on medical institutions sites, such as hospitals and clinics. Patients have the limited access to their personal medical data, while it is difficult to obtain such data from hospitals in real time, or to even share the data with family members and friends. In addition, medical workers in these institutions may access and disclose patients' private medical data at will. Secondly, once a patient and a medical institution have any conflict or dispute, the latter can arbitrarily tamper with the EHRs of the patient, implicitly threatening the patient's case. Thirdly, personal medical records are inherently confidential data and subject to personal privacy and security risks. As such, those records belong only to the corresponding individuals and only authorized users should be able to access relevant information. In order to solve the above problems, some researchers have made significant improvements in enabling patients to generate, manage and share their own EHRs, and ensure the privacy of their medical data.

1.1 Related Works

In recent years, due to the extensive application of cloud computing technology in data processing [1], cloud-based EHRs have developed rapidly and more patients have been able to control their own medical data. In the cloud-based EHRs system, Premarathne et al. [2] and Ramu [3] set the access control to allow patients to share their medical data with doctors in a controlled way. However, a cloud server is not a fully trusted third party. In the cloud storage environment, it is difficult to guarantee the security of EHRs [4].

In 2008, blockchain was first proposed by Nakamoto [5] as a part of cryptocurrency bitcoin. At present, the application of the blockchain technology in the medical field is widely concerned among blockchain researchers [6]. With the decentralized, tamper-proof, traceable and publicly available blockchain technology [7], many problems in the medical field can be solved. Roehrs et al. [8] and Gordon et al. [9] proposed the basic framework of blockchain-based EHRs. Omer et al. [10] and Badr et al. [11] protected sensitive data of patients with the encryption technology in blockchain-based EHRs. Chen et al. [12] proposed a blockchain-based searchable encryption scheme for EHRs, which allowed patients to control the access to their EHRs.

In addition to the encryption protection, the authenticity of EHRs should be considered in the sharing process of EHRs. Authentication is crucial in blockchain [13] and cannot be ignored in blockchain-based EHRs [14]. Considering the characteristics of blockchain, many researchers proposed distributed signature schemes for blockchain-based EHRs. Tang et al. [14] constructed an identity-based signature scheme with multiple authorities to verify the identity of the signer and ensure the authenticity of the EHRs. Guo et al. [15] and Sun et al. [16] designed an attribute-based signature scheme with multiple authorities, which allowed the signer to hide their identity information when signing. However, these signature schemes lacked confidentiality of EHRs.

In order to satisfy both confidentiality and authenticity, in 1997, Zheng [17] first proposed the idea of signcryption, which could simultaneously realize the functions of signing and encrypting plaintext messages. Then, Malone-Lee [18] put forward the first practical identity-based signcryption scheme. Although many researchers have proposed more secure and efficient identity-based signcryption schemes [19,20], these schemes have only considered the case that a message was sent to one receiver. In 2006, Duan et al. [21] first proposed a multi-receiver identity-based signcryption scheme to send the same message to multiple receivers. In their scheme, the sender was only required to perform one pairing computation and n scalar multiplication in the signcryption phase, and each receiver could verify the validity of the message. Since then, the identity-based signcryption scheme for multiple receivers have been significantly improved based on the consideration of the efficiency and security properties [22–24].

In all the above identity-based signcryption schemes, only one key generation center (KGC) generates secret keys for all users, and the users must trust KGC unconditionally. However, KGC can use the public identity of users in the system to calculate the user's secret key. Therefore, it can forge the sender's signcryption or decrypt the signcryption obtained by the receiver. In addition, KGC may face a single point of failure.

1.2 Our Contributions

A centralized KGC will have security risks, and the signcryption scheme of patient's medical data in the blockchain was seldom explored. To enable patients to share the EHRs safely in the blockchain, in this paper, we made the following contributions.

Firstly, the distributed key generation method [13] is introduced into a centralized identity-based signcryption (IBSC) scheme. For multiple receivers, an identity-based signcryption with multiple authorities (MA-IBSC) scheme is developed. N authorities randomly construct their own polynomials and all authorities cooperatively generate the master secret key of the system by secret sharing, and embed their own secret key into the user's secret key. Therefore, the scheme can resist the collusion attack of N-1 corrupt authorities. In addition, after signing EHRs, the patient encrypts the EHRs with the identities of other users whom the patient wants to share the data with. Thus, only authorized receivers can decrypt and access the EHRs. In this way, the authenticity of EHRs is ensured by verifying the signer's signature. Furthermore, in the signcryption process, identity information of receivers can be hidden and the relationships between the patient and receivers are not exposed in the blockchain.

Secondly, signcrypted EHRs are directly uploaded to the blockchain, other nodes cannot verify them. Based on some adjustments of the on-chain and off-chain storage model [13], the signcrypted EHRs are recommended to be stored in the patient's own off-blockchain database, so that the patient can control the EHRs. Then the patient extracts the storage address, signs it with the private key, and uploads it to the blockchain. Other users (nodes) in the system can verify the validity of the given address based on the patient's public key.

Thirdly, based on the assumptions of computational Diffie-Hellman (CDH) problem and Bilinear Computational Diffie-Hellman (BCDH) problem, it is proved that our proposed sigcryption scheme is secure in the random oracle model. In other words, the unforgeability and confidentiality of signcryption are realized. Furthermore, the performance of the scheme is evaluated based on the two indices of signcryption efficiency and signcryption attributes.

The remainder of this paper is organized as follows. Section 2 presents the preliminaries, including Lagrange interpolation, bilinear map, computational assumption, syntax and secure model of the signcryption scheme. In Section 3, the EHRs system model in blockchain is described in detail. Section 4 demonstrates the specific MA-IBSC scheme for multiple receivers. The security analysis and performance evaluation are provided in Section 5. Finally, the conclusion is drawn in Section 6.

2 Preliminaries

2.1 Lagrange Interpolation

For a polynomial f of degree n, given $n+1$ $(x_i, y_i)(i = 1, 2, \ldots, n+1)$ on f, we can uniquely determine a polynomial $f(x) = \sum_{i=1}^{n} y_i \left(\prod_{1 \le j \ne i \le n} \dfrac{x - x_j}{x_i - x_j} \right)$.

2.2 Bilinear Map

Let p be a large prime number, G_1 and G_2 be two multiplicative cyclic groups of order p, and g be the generator of G_1. We say that $e : G_1 \times G_1 \to G_2$ is not a bilinear map unless e satisfies the following properties:

1) Bilinearity: for all $u, v \in G_1$ and $a, b \in Z_p$, $e(u^a, v^b) = e(u, v)^{ab}$;

2) Non-degeneracy: there exists $u, v \in G_1$, such that $e(u, v) \ne 1$. That is to say, mapping e will not map all element pairs in $G_1 \times G_1$ to the identity element of G_2;

3) Computability: for all $u, v \in G_1$, a valid algorithm can be used to calculate $e(u, v)$.

2.3 Computational Assumption

The security of the MA-IBSC scheme for multiple receivers is mainly based on the assumptions of Computational Diffie-Hellman (CDH) problem and Bilinear Computational Diffie-Hellman (BCDH) problem.

1) Computational Diffie-Hellman (CDH) problem. After $a, b \in Z_p^*$ are randomly selected, for the given $g, g^a, g^b \in G_1$, $g^{ab} \in G_1$ is calculated. If there is no probabilistic polynomial time (PPT) adversary A to calculate $g^{ab} \in G_1$ with the probability advantage that cannot be ignored, we call CDH in group G_1 the assumption of the difficult problem.

2) Bilinear Computational Diffie-Hellman (BCDH) problem. After $a, b, c \in Z_p^*$ are randomly selected, for the given $g, g^a, g^b, g^c \in G_1$, $e(g, g)^{abc} \in G_2$ is calculated. If there is no probabilistic polynomial time (PPT) adversary A to calculate $e(g, g)^{abc} \in G_2$ with the probability advantage that cannot be ignored, we call BCDH in group G_1 the assumption of the difficult problem.

2.4 Syntax of the Signcryption Scheme

The identity-based signcryption with multiple authorities (MA-IBSC) scheme for multiple receivers involves the following seven algorithms:

Global Setup: The EHRs server takes a security parameter λ as the input and then outputs system public parameters *params*.

Authority Setup: All authorities perform this algorithm interactively. They input public parameters *params* and their identity ID_i, then generate their respective secret key sk_{ID_i}, system master secret key s and master public key PK.

KeyGen: This algorithm is also cooperatively controlled by all authorities. They input the public parameters *params*, their respective secret key sk_{ID_i}, and identity id_i of a user, and then return secret key sk_{id_i} to the user.

User-Sign: User id_i takes public parameters *params*, his/her secret key sk_{id_i} and message M as input to run this algorithm with, and then outputs the signature σ_i of M.

User-Encrypt: User id_i usually executes this algorithm after the User-Sign algorithm. User id_i inputs the public parameters *params*, the signature σ_i of M, and the public keys of the receivers, and then outputs the signcryption message SC of M.

Verify: To verify the signature σ_i of M, other users take the signer's identity id_i, M and σ_i as input to carry out this algorithm. If the signature σ_i is valid, it returns *Accept*, otherwise returns *Reject*.

Receiver-Decrypt: Only the receivers picked by the user can run the algorithm to decrypt SC. Any one of the receivers inputs public parameter *params*, SC, and the user's secret key to the algorithm, and then obtains M and the sharer's id_i.

2.5 Security Model

Definition 1 and Definition 2 respectively introduce the two security attributes of the adapted signcryption scheme: unforgeability and confidentiality.

Definition 1: Suppose F is a forger, $\mho$ is defined as the MA-IBSC scheme for multiple receivers. The game between F and Challenger C is described as follows:

Global Setup: Challenger C takes a security parameter λ as input, runs global setup algorithm, then generates *params* and transmits it to F.

Authority Setup: Challenger C runs authority setup algorithm to output secret key sk_{ID_i} for each authority ID_i, where $i \in \{1, 2, \ldots N\}$. Then Forger F outputs his/her target identity id_{i^*}.

Queries: Forger F performs the following four queries to Challenger C:

- Secret key queries: F asks C for the secret key of some authorities $ID_{i\in Q_S}$, where $Q_S \subset \{1, 2, \ldots, N\}$ represents the index set of corrupt authorities, and then Challenger C outputs $sk_{ID_{i\in Q_S}}$ to F.
- Key generation queries: When C receives the private key query about identity id_i, C runs the key generation algorithm and returns sk_{id_i} to F.
- User-sign queries: When C receives the signature query about message M and identity id_i, C returns σ_i to F.
- User-encrypt queries: To forge a signcryption, the user-encrypt query always follows user-sign query. When C receives the encryption query about (M, R, id_i), where R represents the identity set of the receivers, namely, $R = \{id_l\}_{l=1}^{n}$, then C calculates signcryption SC and returns it to F.

Forgery: Forger F finally outputs a new signcryption SC^* and the public key pair $(id_1, sk_{id_1}), (id_2, sk_{id_2}), \ldots, (id_n, sk_{id_n})$ of n receivers. If SC^* is the signcryption of id_{i^*} to the message M and can be correctly decrypted and verified by receivers in set $R = \{id_l\}_{l=1}^{n}$, then SC^* is a valid signcryption and F wins the game. The limitations here are described below. F cannot query the $sk_{id_{i^*}}$ with identity id_{i^*} through the key generation query, and SC^* cannot be generated by the User-Sign and User-Encrypt algorithm.

Definition 2: Suppose that A is an adversary, $\mho$ is defined as the MA-IBSC scheme for multiple receivers. The game between Adversary A and Challenger C is introduced as follows:

Global Setup: Challenger C takes a security parameter λ as input, runs global setup algorithm, and then generates *params* and transmits it to A.

Authority Setup: Challenger C runs authority setup algorithm to output secret key sk_{ID_i} for each authority ID_i, where $i \in \{1, 2, \ldots, N\}$. Adversary A outputs target identities id_{l^*} of n receivers, where $l = \{1, 2, \ldots, n\}$.

Phase 1: Adversary A performs the following five queries to Challenger C:

- Secret key queries: A asks C for the secret key of some authorities $ID_{i\in Q_S}$, where $Q_S \subset \{1, 2, \ldots, N\}$ represents the index set of corrupt authorities, and then Challenger C outputs $sk_{ID_{i\in Q_S}}$ to A.
- Key generation queries: When C receives the private key query about identity id_i, C runs the key generation algorithm and returns sk_{id_i} to A.
- User-sign queries: When C receives the signature query about message M and identity id_{i^*}, where id_{i^*} is the user being attacked, then C returns σ_i to A.
- User-encrypt queries: The user-encrypt query always follows the user-sign query. When C receives the encryption query about (M, R, id_{i^*}), where R represents the identity set of the receivers, namely $R = \{id_l\}_{l=1}^{n}$, then C calculates signcryption SC and returns to A.
- Receiver-Decrypt-and-Verify queries: When C receives the decryption and verify query together about (SC, id_l, id_{i^*}), where $id_l \in R$, if SC is a valid singcryption, then C decrypts it, verifies M, and returns M to A.

Challenge: A outputs a target plaintext pair (M_0, M_1) and a private key $sk_{id_{i^*}}$. When Challenger C receives (M_0, M_1) and $sk_{id_{i^*}}$, C randomly selects a message M_β, where $\beta \in \{0, 1\}$, then generates the target signcryption SC^* based on M_β, $sk_{id_{i^*}}$ and n target receivers id_{l^*}, where $l = \{1, 2, \ldots, n\}$, and finally returns SC^* to A.

Phase 2: A makes multiple queries as those in Phase 1. The limitations here are described below. A cannot ask $sk_{id_{l^*}}$ of n target receivers id_{l^*}, where $l = \{1, 2, \ldots, n\}$ during the key generation query, and A cannot ask SC^* during Receiver-Decrypt-and-Verify query.

Guess: In the end, A outputs its guess $\beta' \in \{0, 1\}$ and wins the game if $\beta' = \beta$.

3 EHRs System Model in Blockchain

In this section, the EHRs system model in blockchain is introduced in detail. The model combines the EHRs system with the MA-IBSC scheme for multiple receivers, realizes the sharing of EHRs in the blockchain, and ensures the privacy and validity of EHRs. The system roles, EHRs storage mode, authentication cases, and the application of the signcryption scheme are introduced below.

3.1 System Roles

There are three main roles in EHRs system in the blockchain: EHRs server, authority, and user.

EHRs Server: The EHRs server is mainly responsible for generating public parameters *params* in EHRS system initialization, and distributing corresponding identity for each authority and each user in the system.

Authority: The authorities include all medical departments: hospitals, pharmacies, health insurance companies, medical research institutes and so on. As the bookkeeping nodes in the blockchain, they package a set of transactions that are broadcast on the network and upload them to the new block created by them through the DPoS consensus mechanism.

User: As ordinary nodes in the blockchain, users primarily create new transactions and publish them to the network. Users include patients, medical workers and common people. Patients create their own EHRs after treatment, and then adopt MA-IBSC scheme to share their private EHRs with other designated users in the blockchain.

3.2 EHRs Storage Mode

EHRs of patients are generally private data and cannot be directly uploaded to the blockchain for sharing. Therefore, we adopt the on-chain and off-chain storage mode and only upload the address of the stored EHRs to the blockchain. The EHRs are signcrypted and stored in the off-blockchain database of each node, and the decryption permission is set at the same time. This storage mode enables patient's EHRs to be safely shared among the users that the patient designates.

As shown in Fig. 1, when a patient creates his/her own new EHRs after diagnosis or treatment, he/she uses his/her secret key and the public keys of users, whom he/she wants to share the data with, to signcrypt the EHRs, and stores the signcrypted data in his/her off-blockchain database. Then he/she signs the address of the stored EHRs and publishes it to the blockchain.

3.3 Authentication Cases

To guarantee that the EHRs shared by the patient and the storage address of the EHRs broadcast by the patient in the blockchain are real, it is necessary to perform authentication. Authentication is mainly performed by verifying the signature of the sharer. Based on the system model and EHRs storage mode, authentication can be mainly classified into the following two cases:

- Case 1 (Signature Authentication): Only the address of the stored EHRs is uploaded to the blockchain. Therefore, the patient needs to sign it with his/her own secret key, and other users can verify the authenticity and validity of the address.

- Case 2 (Signcryption Authentication): All users can retrieve the patient's signcrypted EHRs with the address stored in the blockchain. However, only the users (such as doctors, family members, and friends) authorized by the patient can decrypt the EHRs with their secret key, and then verify the signature of the patient to ensure the authenticity of the patient's identity and the EHRs.

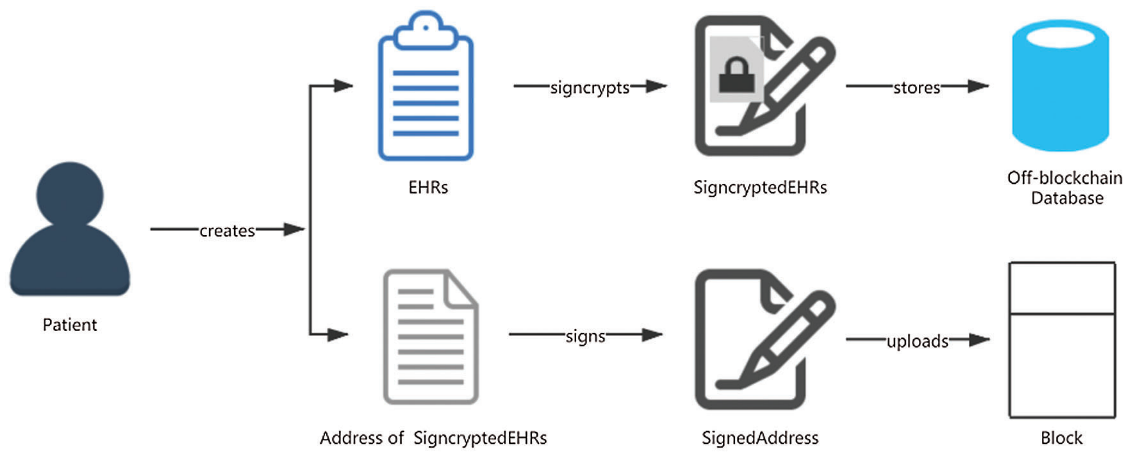

Figure 1: Storage Mode of EHRs in Blockchain

3.4 Application of the Sigcryption Scheme

For the purposes of realizing the signcryption of EHRs and the two authentication cases, we describe the relationships between the system roles and the MA-IBSC scheme for multiple receivers below.

First, the EHRs server runs the **Global Setup** algorithm to generate the public parameters of the system. Next, each authority performs **Authority Setup** algorithm to produce its own secret key and then cooperates with other authorities to generate the master secret key and the master public key of the system. After that, with the identity of each user in the system, each authority runs the **KenGen** algorithm and jointly distribute the secret key to the user. After receiving the secret key, the patient uses **User-Sign** algorithm to sign his/her own EHRs, executes **User-Encrypt** algorithm immediately, encrypts the signed EHRs with the public keys of the receivers whom he/she wants to share the data with. In this way, the decryption permission is set for these designated receivers. After storing the signcrypted EHRs in the off-blockchain database, the patient executes the **User-Sign** algorithm again and broadcasts the signed storage address of EHRs. All other nodes (authorities or users) can verify the validity of the address given by the patient by executing the **Verify** algorithm. Then, for a period of time, the bookkeeping node packs the storage addresses of EHRs signed by some patients, and uploads them to a new block, which is connected by the hash value of the previous block to form a blockchain. The data structure of blockchain is shown in Fig. 2.

When other users want to access the patient's EHRs, they retrieve the patient's signcrypted data in the off-blockchain database through the storage address on the blockchain and then run the **Receiver-Decrypt** algorithm. Only receivers with the decryption permission set by the patient can decrypt the signcrypted EHRs with their secret keys, and then run the **Verify** algorithm to ensure that the real EHRs are obtained.

4 Proposed Signcryption Scheme

Based on the EHRs system of blockchain, we propose an identity-based signcryption with multiple authorities (MA-IBSC) scheme for multiple receivers. In the scheme, users are issued their secret keys from N authorities. In addition, a user can send the same signcryption information to multiple receivers. The anonymity of the receivers is realized by Lagrange interpolation.

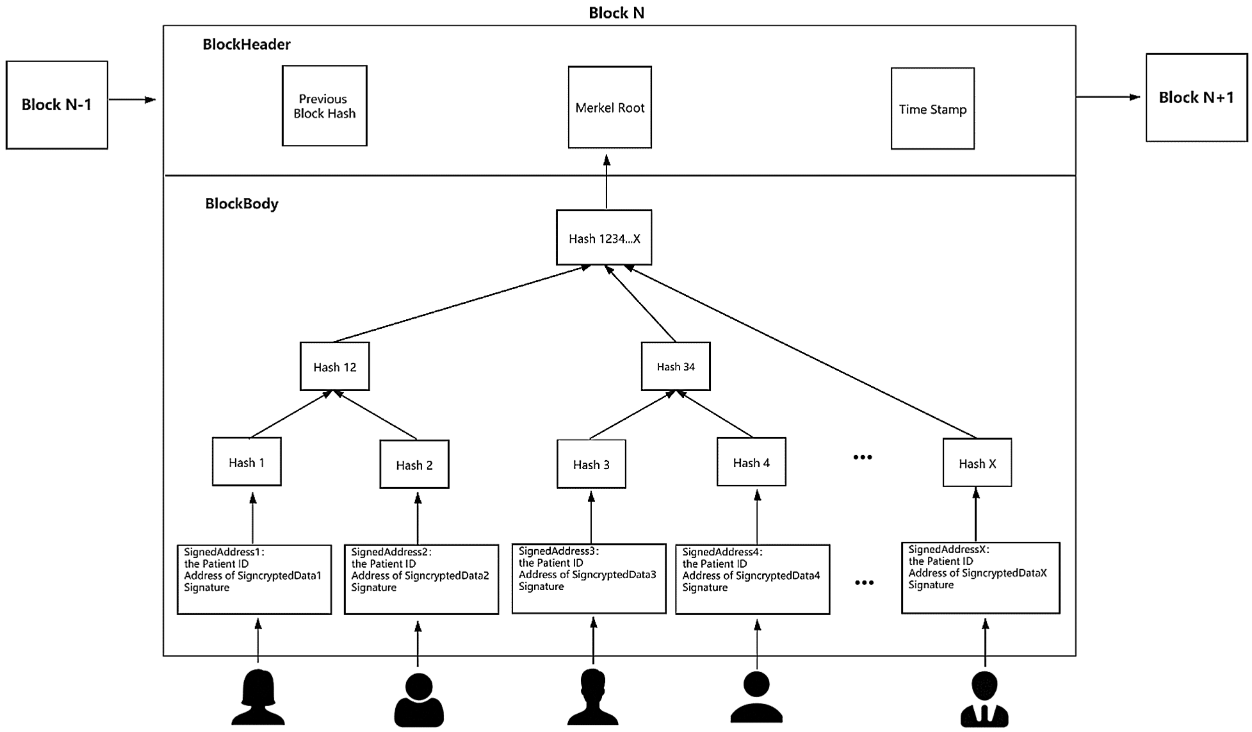

Figure 2: Data Structure of Blockchain in EHRs System

The detailed MA-IBSC scheme for multiple receivers is introduced bellow:

Global Setup: The EHRs server chooses two suitable multiplicative cyclic groups G_1 and G_2 with a prime order p, equipped with a bilinear map $e : G_1 \times G_1 \rightarrow G_2$. Assuming that g is a random generator of G_1, an element P_1 in G_1 is randomly selected. There are four strong collision-resistant hash functions $H_0 : \{0,1\}^{\lambda_1} \rightarrow Z_p^*$, $H_1 : \{0,1\}^{\lambda_1} \rightarrow G_1$, $H_2 : G_1 \times \{0,1\}^{\lambda_2} \rightarrow Z_p^*$, and $H_3 : G_2 \rightarrow \{0,1\}^{\lambda_1+\lambda_2}$, where λ_1 and λ_2 represent the length of each user's identity and the length of message, respectively. Suppose that there are N authorities in the system. The public parameters of the system are $params = \langle p, g, P_1, e, G_1, G_2,\ H_0, H_1, H_2, H_3, N \rangle$.

Authority Setup: Each authority runs this algorithm with the input of public parameters *params* and identity ID_i, where $i \in \{1, 2, \ldots, N\}$. The two phases of generating master secret key s, master public key PK and authority's secret key sk_{ID_i} , where $i \in \{1, 2, \ldots, N\}$, are described as follows:

- Phase 1 (generation of the master secret key of the system and the secret key of each authority):

1) First, each authority ID_i randomly selects a polynomial P(x) of $N-1$ degree over Z_p^*:

$$P(x) = \left(a_{i0} + a_{i1}x + \ldots + a_{i(N-1)}x^{N-1}\right)(mod\ p) \tag{1}$$

To hide the polynomial coefficients, $A_{ik} = g^{a_{ik}}$ is calculated and broadcast, where $k = (0, 1, \ldots, N-1)$. Second, it calculates secret shares $s_{ij} = P\left(H_0\left(ID_j\right)\right)$, where $j = (1, 2, \ldots, N)$. Finally, it secretly sends s_{ij} to ID_j for $j \neq i$.

2) After receiving the secret share s_{ij} from ID_i, each authority ID_j verifies whether the equation $g^{s_{ij}} = \prod_{k=0}^{N-1} (A_{ik})^{H_0\left(ID_j\right)^k}$ holds. If it holds, the secret share s_{ij} is valid and the sender ID_i is considered to

be honest. If not, ID_j broadcasts a complaint against ID_i. Then, to prove its honesty, ID_i needs to keep broadcasting the secret shares s_{ij} until the equation holds.

3) After the above interactions, N authorities jointly generate the master secret key $s = \sum_{i=1}^{N} a_{i0}$. If the number of corrupt authorities is less than N, they cannot recover s. The secret key of each authority ID_i is the constant term of its randomly selected polynomial, namely, $sk_{ID_i} = a_{i0}$, where $i \in \{1, 2, \ldots, N\}$.

- Phase 2 (generation of the master public key of system): In Phase 1, each authority has broadcast a publicly verifiable value $\{A_{i0} = g^{a_{i0}}\}$, where $i \in \{1, 2, \ldots, N\}$. Thus, the master public key PK is calculated as:

$$PK = \prod_{i=1}^{N} A_{i0} = \prod_{i=1}^{N} g^{a_{i0}} = g^{\sum_{i=1}^{N} a_{i0}} = g^{s} \in G_1 \tag{2}$$

Finally, each authority adds parameters $\{(ID_i, A_{i0})\}_{i=1}^{N}$ and PK to $params$, which is finally expressed as:

$$params = \left\langle p, g, P_1, e, G_1, G_2, H_0, H_1, H_2, H_3, N, \{(ID_i, A_{i0})\}_{i=1}^{N}, PK \right\rangle \tag{3}$$

KeyGen: When a user with his/her identity id_i registers in the EHRs system of blockchain, he/she obtains his/her public key pk_{id_i} and secret key sk_{id_i} from N authorities. The process consists of the following three phases.

- Phase 1 (generation of the public key and partial secret key): First, every authority ID_j, where $j \in \{1, 2, \ldots, N\}$, calculates the user's public key $pk_{id_i} = H_1(id_i)$ with his/her identity id_i, then calculates partial secret key $psk_{id_i,j} = (pk_{id_i} \cdot P_1)^{a_{j0}}$ and secretly sends it to id_i.
- Phase 2 (verification of the partial secret key): After receiving the $psk_{id_i,j}$ from authority ID_j, id_i verifies whether the equation $e(psk_{id_i,j}, g) = ?e(pk_{id_i}, A_{j0})$ holds. If it holds, the partial secret key is valid. If not, the authority ID_j needs to transmit the partial secret key again until the equation holds.
- Phase 3 (generation of the secret key): Through the above interactions, user id_i receives all partial secret keys from N authorities, and then calculates his/her secret key sk_{id_i} as:

$$sk_{id_i} = \prod_{j=1}^{N} psk_{id_i,j} = \prod_{j=1}^{N} (pk_{id_i} \cdot P_1)^{a_{j0}} = (pk_{id_i} \cdot P_1)^{s} \tag{4}$$

User-Sign: To sign a message M, user (mainly refers to the patient user in the system) id_i selects a random integer $r \in Z_q^*$, and then calculates $X = g^r$, $h = H_2(M, X)$ and $W = {sk_{id_i}}^{h} {pk_{id_i}}^{r}$. The signature σ_i of message M is $\sigma_i = (X, W)$.

User-Encrypt: To complete the signcryption of M, this algorithm is usually used after the **User-Sign** algorithm. Encryption is mainly divided into the following six steps. First, user id_i calculates $V = e(PK^r, P_1)$, $Z = H_3(V) \oplus (id_i||M)$. Second, he/she selects other users whom he/she wants to share message M with, counts the number n of these receivers, calculates $x_l = H_0(id_l)$ and $y_l = {pk_{id_l}}^{r}$ based on the identity id_l of the n receivers, where $l = (1, 2, \ldots, n)$, and then gets n sets of data: (x_1, y_1), (x_2, y_2), ... , (x_n, y_n). Third, $n-1$ degree polynomial $F(x)$ is constructed by Lagrange interpolation, so that $F(x_l) = y_l$, where $l \in \{1, 2, \ldots, n\}$. Fourth, for $l = (1, 2, \ldots, n)$, the user id_i calculates

$$f_l(x) = \prod_{1 \le m \ne l \le n} \frac{x - x_m}{x_l - x_m} = b_{l,1} + b_{l,1}x + \ldots + b_{l,n}x^{n-1} \tag{5}$$

After that, for $l = (1, 2, \ldots, n)$, $L_l = \sum_{m=1}^{n} b_{m,l}y_m$ is calculated. Finally, the signcryption of M is expressed as:

$$SC = \langle L_1, L_2, \ldots, L_n, \sigma_i = (X, W), Z \rangle \tag{6}$$

As you can see, the identity information of receivers is not directly displayed in the SC.

Verify: To verify the validity of signature σ_i from user id_i, first, other users calculate $h = H_1(M, X)$, and then verify whether the equation $e(g, W) = ?e\left(PK^h \cdot X, pk_{id_i}\right) \cdot e\left(PK^h, P_1\right)$ holds or not. If it holds, the signature from id_i is valid and it returns *Accept*. If not, it returns *Reject*.

Receiver-Decrypt: Only the receiver with identity $\{id_l\}_{l=1}^n$ designated by sharer id_i has the right to decrypt the signcryption SC and obtain message M. The receiver id_l takes SC, *params*, his/her identity id_l and secret key sk_{id_l} as inputs to run this algorithm. He/she first calculates $x_l = H_0(id_l)$, $\delta_l = L_1 + x_l L_2 + \ldots + ({x_l}^{n-1} mod p) L_n$, and $V' = \frac{e(X, sk_{id_l})}{e(PK, \delta_l)}$ and then gets the message M and the identity id_i of the signer through the following calculation:

$$H_3(V') \oplus Z = (id_i || M) \tag{7}$$

Correctness:

1) The correctness of signature σ_i from user id_i is derived from the following equation:

$$\begin{aligned} e(g, W) &= e\left(g, {sk_{id_i}}^h {pk_{id_i}}^r\right) \\ &= e\left(g, (pk_{id_i} \cdot P_1)^{sh} {pk_{id_i}}^r\right) \\ &= e\left(g, {pk_{id_i}}^{sh+r}\right) \cdot e\left(g, {P_1}^{sh}\right) \\ &= e\left(g^{sh} g^r, pk_{id_i}\right) \cdot e\left(g^{sh} \cdot P_1\right) \\ &= e\left(PK^h \cdot X, pk_{id_i}\right) \cdot e\left(PK^h, P_1\right) \end{aligned} \tag{8}$$

2) When $V' = V$, message M of the user id_i can be obtained. For each $l \in \{1, 2, \ldots, n\}$, there is $y_l = {pk_{id_l}}^r$. According to Lagrange interpolation, we can calculate:

$$\begin{aligned} \delta_l &= L_1 + x_l L_2 + \ldots + {x_l}^{n-1} L_n \\ &= \left(b_{1,1} y_1 + \ldots + b_{n,1} y_n\right) + \left(x_l b_{1,2} y_1 + \ldots + x_l b_{n,2} y_n\right) + \ldots + \left({x_l}^{n-1} b_{1,n} y_1 + \ldots + {x_l}^{n-1} b_{n,n} y_n\right) \\ &= \left(b_{1,1} + b_{1,2} x_l + \ldots + b_{1,n} {x_l}^{n-1}\right) y_1 + \left(b_{2,1} + b_{2,2} x_l + \ldots + b_{2,n} {x_l}^{n-1}\right) y_2 + .. \\ &\quad + \left(b_{n,1} + b_{n,2} x_l + \ldots + b_{n,n} {x_l}^{n-1}\right) y_n \\ &= y_l = {pk_{id_l}}^r \end{aligned} \tag{9}$$

Thus, the correctness of decryption is derived from the following two equations:

$$\begin{aligned} V' &= \frac{e(X, sk_{id_l})}{e(PK, \delta_l)} \\ &= \frac{e(g^r, (pk_{id_l} \cdot P_1)^s)}{e(g^s, y_l)} \\ &= \frac{e(g^s, {pk_{id_l}}^r) \cdot e(g^s, {P_1}^r)}{e(g^s, {pk_{id_l}}^r)} \\ &= e(PK, P_1)^r = V \end{aligned} \tag{10}$$

and

$$H_3(V') \oplus Z = H_3(V) \oplus Z = (id_i || M) \tag{11}$$

5 Security Analysis and Performance Evaluation

5.1 Security Proof

In this section, Theorem 1 and Theorem 2 respectively prove the unforgeability and confidentiality of signcryption.

The master secret key s is randomly generated by N authorities by the distributed key generation and no one knows the real value of s, so g^s cannot be used as an instance of CDH problem. Here, we set $PK = g^{as}$, where g^a is a CDH instance. s is still generated by all authorities randomly and unknown to others, a and s are independent of each other. For any PPT adversary, even if he/she corrupts $N - 1$ authorities, he/she cannot recover the value s. Therefore, for the PPT adversary, $PK = g^s \in G_1$ and $PK = g^{as} \in G_1$ are indistinguishable.

Theorem 1: In the random oracle model, if there is a probabilistic polynomial time (PPT) adversary F, who can win the **Definition 1** game in Section 2.5 with a non-negligible advantage ε within time τ, then there is an algorithm C that can solve the CDH problem with the advantage $\varepsilon' \geq \varepsilon - \frac{Q_{H_2}(Q_{US} + Q_{UE})}{2^k}$ within time $\tau' \approx \tau + Q_{UE}O(\tau_1)$, where τ_1 is the running time of e. (PPT adversary can make Q_S secret key quires, Q_K key generation quires, Q_{US} user-sign quires, Q_{UE} user-encrypt quires and Q_{H_0}, Q_{H_1}, Q_{H_2}, Q_{H_3} hash function H_0, H_1, H_2, H_3 quires at most).

Proof: The following shows how algorithm C uses F to solve the CDH problem with probability ε' within time τ'.

First, C gets an instance $\langle G_1, G_2, p, g, e, g^a, g^b \rangle$ of CDH problem, whose goal is to calculate $g^{ab} \in G_1$. C simulates a challenger to play the following game with F.

Global Setup: Challenger C executes global setup algorithm, inputs parameter λ, outputs public parameter *params* and sends it to F.

Authority Setup: C represents all authorities to run the authority setup algorithm and generate secret key sk_{ID_i} for each authority ID_i, where $i \in \{1, 2, \ldots N\}$, so only C knows the real value s of the master secret key. However, C sets the master secret key as $a \cdot s$, and sets the public key as $PK = g^{as}$. Because s and $a \cdot s$ are unknown to F, g^s and g^{as} are indistinguishable to F. Finally, C adds parameters $\{(ID_i, A_{i0})\}_{i=1}^{N}$ and $PK = g^{as}$ to *params*. F can obtain $params = \left\langle p, g, P_1, e, G_1, G_2, H_0, H_1, H_2, H_3, N, \{(ID_i, A_{i0})\}_{i=1}^{N}, PK \right\rangle$ from C. After receiving the *params*, F outputs the target identity id_{i^*}.

H_0, H_1, H_2, and H_3 are random oracle models controlled by C. The query results of H_0, H_1, H_2, and H_3 are stored in $H_0 - list$, $H_1 - list$, $H_2 - list$, and $H_3 - list$ respectively.

Queries: Forger F performs some queries to Challenger C:

- H_0 queries: C enters an identity ID_i or id_i into H_0. If there is (ID_i, x_i) or (id_i, x_i) in the $H_0 - list$, returns x_i, otherwise C performs the following steps:

1) Randomly selects an integer $x_i \in Z_p^*$;

2) Saves (ID_i, x_i) or (id_i, x_i) to $H_0 - list$;

3) Returns x_i.

- H_1 queries: C enters an identity id_i into H_1. If there is (id_i, k_i, pk_{id_i}) in the $H_1 - list$, returns pk_{id_i}, otherwise C performs the following steps:

1) Randomly selects an integer $k_i \in Z_p^*$;

2) If $id_i = id_{i^*}$, (where id_{i^*} is C random guess of the identity that F will attack) calculates $\frac{g^{k_i}}{P_1}$, otherwise calculates g^{k_i};

3) Saves (id_i, k_i, pk_{id_i}) to $H_1 - list$;

4) Returns pk_{id_i}.

- H_2 queries: C enters an array (M_i, X_i) into H_2. If there is (M_i, X_i, h_i) in the $H_2 - list$, returns h_i, otherwise C performs the following steps:

1) Randomly selects an integer $h_i \in Z_p^*$;

2) Saves (M_i, X_i, h_i) to $H_2 - list$;

3) Returns h_i.

- H_3 queries: C enters an element $V_i \in G_2$ into H_3. If there is (V_i, ρ_i) in the $H_3 - list$, returns ρ_i otherwise C performs the following steps:

1) Randomly selects a character string $\rho_i \in \{0, 1\}^{\lambda_1+\lambda_2}$;

2) Saves (V_i, ρ_i) to $H_3 - list$;

3) Returns ρ_i.

- Secret key queries: F requests secret keys $\mathrm{sk}_{ID_{i\in Q_S}}$ of authority $\mathrm{ID}_{i\in Q_S}$, where $Q_S \subset \{1, 2, \ldots, N\}$ represents the index set of corrupt authorities. Because C generates the secret keys of all authorities , C can answer the queries from F.

- Key generation queries: F asks C about the secret key sk_{id_i} of the identity id_i. If $id_i = id_{i^*}$, C does not answer this query and terminate the game. Otherwise, C looks for (id_i, k_i, pk_{id_i}) in $H_1 - list$, calculates $sk_{id_i} = \left(g^{k_i} \cdot P_1\right)^{as}$, and then returns it to F.

- User-sign queries: F asks C for the signature σ_i of a tuple (id_i, M). If $id_i \neq id_{i^*}$, C will get the correct sk_{id_i} from key generation queries, and then calculates the signature σ_i and transmits it to F. If $id_i = id_{i^*}$, C cannot obtain $sk_{id_{i^*}}$ from key generation queries to calculate the signature directly. However, C can answer F's query through the following steps: 1) C randomly selects $r' \in Z_p^*$ and calculates $X' = g^{r'}$. 2) C finds (M, X', h') in $H_2 - list$ list and gets h'. 3) C finds (id_i, k_i, pk_{id_i}) in $H_1 - list$ (if it cannot be found, C chooses $k_i \in Z_p^*$, then calculates $pk_{id_i} = \frac{g^{k_i}}{P_1}$ and stores (id_i, k_i, pk_{id_i}) in $H_1 - list$). 4) C calculates $\mathrm{W}' = {sk_{id_i}}^{h'} {pk_{id_i}}^{r'} = \frac{g^{k_i(as\cdot h'+r')}}{{P_1}^{r'}}$, and then gets $\sigma_i = (X', W')$ and returns it to F.

- User-encrypt queries: To forge a signcryption, the query is executed after the user-sign query. When C receives the encryption query about (M, R, id_i), where $id_i = id_{i^*}$ and R represents a receiver set $\{id_l\}_{l=1}^n$ (l represents the identity of receivers and n represents the number of receivers), C answers F through the following steps: 1) C calculates $V' = e\left(PK^{r'}, P_1\right) = e\left(g^{as\cdot r'}, P_1\right)$, and then finds (V', ρ') in the $H_3 - list$. 2) C calculates $Z' = \rho' \oplus (id_i||M)$; 3) C finds (id_l, x_l) in the $H_0 - list$, calculates $y_l = {pk_{id_l}}^{r'}$ and gets L_l, where $l \in \{1, 2, \ldots, n\}$. 4) C gets the signcryption SC and sends it to F.

Forgery: F generates the target signcryption:

$$SC^* = \langle {L_1}^*, {L_2}^*, \ldots, {L_n}^*, {\sigma_i}^* = (X^*, W^*), Z^* \rangle \tag{12}$$

If the forgery is successful, the following equation holds:

$$e(g, W^*) = ?e\left(PK^h \cdot X^*, pk_{id_i}\right) \cdot e\left(PK^h, P_1\right) \tag{13}$$

Define $b = k_i h$, then $W^* = {sk_{id_i}}^h {pk_{id_i}}^r = \left(g^{as \cdot k_i}\right)^h {pk_{id_i}}^r = g^{ab \cdot s} {pk_{id_i}}^r$. Therefore, we can get the solution of CDH problem $g^{ab} = \left(\frac{W^*}{{pk_{id_i}}^r}\right)^{s^{-1}}$.

In the general signcryption query, as most Q_{H_2} H_2 queries are conducted, the probability that C fails to answer a signcryption query is not greater than $\frac{Q_{H_2}(Q_{US} + Q_{UE})}{2^k}$. Therefore, C can get the advantage $\varepsilon' \geq \varepsilon - \frac{Q_{H_2}(Q_{US} + Q_{UE})}{2^k}$ and $\tau' \approx \tau + Q_{UE} O(\tau_1)$, where τ_1 is the running time of e. From the above proof and CDH problem, we can see that this scheme satisfies the unforgeability of signcryption.

Theorem 2: In the random oracle model, if there is a probabilistic polynomial time (PPT) adversary A, who can win the **Definition 2** game in Section 2.5 with a non-negligible advantage ε within time τ, then there is an algorithm C that can solve the BCDH problem with the advantage $\varepsilon' \geq \varepsilon - \frac{Q_{H_3} Q_{RD\&V}}{2^k}$ within time $\tau' \approx \tau + (2Q_{RD\&V} + Q_{UE}) O(\tau_1)$, where τ_1 is the running time of e. (PPT adversary can make Q_S secret key quires, Q_K key generation quires, Q_{US} user-sign quires, Q_{UE} user-encrypt quires, $Q_{RD\&V}$ receiver-decrypt-and-verify quires and Q_{H_0}, Q_{H_1}, Q_{H_2}, and Q_{H_3} hash function H_0, H_1, H_2, and H_3 quires at most).

Proof: The following shows how algorithm C uses A to solve the BCDH problem with probability ε' within time τ'.

First, C gets an instance $\langle G_1, G_2, p, g, e, g^a, g^b, g^c \rangle$ of BCDH problem, whose goal is to calculate $e(g,g)^{abc} \in G_2$. C simulates a challenger to play the following game with A.

Global Setup: Challenger C executes global setup algorithm, inputs parameter λ, outputs public parameter *params* and sends it to F.

Authority Setup: C represents all authorities to run authority setup algorithm and generate secret key sk_{ID_i} for each authority ID_i, where $i \in \{1, 2, \ldots N\}$. Similarly, C sets $PK = g^{as}$ instead of $PK = g^s$, where g^{as} and g^s are indistinguishable to A. Finally, C adds parameters $\{(ID_i, A_{i0})\}_{i=1}^N$ and $PK = g^{as}$ to *params*. A can obtain $params = \left\langle p, g, P_1, e, G_1, G_2, H_0, H_1, H_2, H_3, N, \{(ID_i, A_{i0})\}_{i=1}^N, PK \right\rangle$ from C. After receiving the *params*, A outputs target identities id_{l^*} of n receivers, where $l = \{1, 2, \ldots, n\}$.

Phase 1: Adversary A performs the following five queries to Challenger C:

- Secret key queries: A requests secret keys $\mathrm{sk}_{ID_{i \in Q_S}}$ of authority $\mathrm{ID}_{i \in Q_S}$, where $Q_S \subset \{1, 2, \ldots, \mathrm{N}\}$ represents the index set of corrupt authorities. Because C generates the secret keys of all authorities , C can answer the queries from A.

- Key generation queries: A asks C about the secret key sk_{id_i} of the identity id_i. If $id_i = id_{l^*}$, where $l = \{1, 2, \ldots, n\}$, C does not answer this query and terminate the game. Otherwise, C looks for (id_i, k_i, pk_{id_i}) in $H_1 - list$, then calculates $sk_{id_i} = (pk_{id_i} \cdot P_1)^{as} = \left(\frac{g^{k_i}}{P_1} \cdot P_1\right)^{as} = g^{as \cdot k_i}$, and returns it to A.

- User-sign queries: A asks C about the signature σ_i of a tuple (id_i, M), where $id_i \neq id_{l^*}$ $(l \in \{1, 2, \ldots, n\})$. C answers A through the following calculations: 1) C randomly selects $r', h, t \in Z_p^*$, calculates $X = \frac{g^{r'}}{g^{as \cdot h}}$, $W = (\frac{g^{k_i}}{P_1})^{r'} \cdot {P_1}^{as \cdot h}$, $P_1 = g^t$, and gets (M, X, h). 2) C finds (M, X) in

$H_2 - list$ so that it does not appear in $H_2 - list$. Otherwise, C reselects $r', h, t \in Z_p^*$, repeats the above calculation step, and then adds eligible (M, X) to $H_2 - list$. 3) C gets $\sigma_i = (X, W)$ of id_i and returns it to A.

- User-encrypt queries: To form a complete signcryption, the query is executed after the user-sign query. When C receives the encryption query about (M, R, id_i), where $id_i \neq id_{l^*}$ and R represents a set of n receivers $\{id_l\}_{l=1}^n$, C answer A through the following steps: 1) C calculates $V = e(PK^{r'}, P_1) = e(PK^{r'}, g^t)$, and then finds (V, ρ) in the $H_3 - list$. 2) C calculates $Z = \rho \oplus (id_i || M)$. 3) C finds (id_l, x_l) in the $H_0 - list$, calculates $y_l = X^{(k_l - t)}$ and gets L_l, where $l \in \{1, 2, \ldots, n\}$. 4) C gets the signcryption SC and sends it to A.

- Receiver-Decrypt-and-Verify queries: When C receives the decrypt-and-verify query about a signcryption $SC = \langle L_1, L_2, \ldots, L_n, \sigma_i = (X, W), Z \rangle$ and an identity id_l, where $l \in \{1, 2, \ldots, n\}$, C answers A through the following steps: 1) C finds (id_l, x_l) in the $H_0 - list$ and calculates $\delta_l = L_1 + x_l L_2 + \ldots + x_l{}^{n-1} L_n$. 2) C finds (id_l, k_l, pk_{id_l}) in the $H_1 - list$, then calculates $sk_{id_l} = PK^{k_l} = g^{as \cdot k_l}$ and $V' = \dfrac{e(X, sk_{id_l})}{e(PK, \delta_l)}$, so C can obtain $(id_i || M) = H_3(V') \oplus Z$. 3) C finds (id_i, k_i, pk_{id_i}) in $H_1 - list$ and gets pk_{id_i}. 4) C verifies that $e(g, W) = ?e(PK^h \cdot X, pk_{id_i}) \cdot e(PK^h, P_1)$ holds. If it holds, SC is a valid signcryption and M is returned to A.

Challenge: A selects a target plaintext pair (M_0, M_1) and identity id_i of the same signer and encryptor. When Challenger C receives (M_0, M_1) and id_i, C randomly selects a message M_β to signcrypt, where $\beta \in \{0, 1\}$. The signcryption calculation is as follows: 1) C finds $(id_{l^*}, k_{l^*}, pk_{id_{l^*}})$ in $H_1 - list$, where $l = \{1, 2, \ldots, n\}$, and then obtains their $pk_{id_{l^*}} = \dfrac{g^{k_{l^*}}}{P_1}$. 2) C calculates $y_{l^*} = pk_{id_{l^*}}{}^r$ and gets L_{l^*}, where $l = \{1, 2, \ldots, n\}$. 3) C generates the target signcryption $SC^* = \langle L_1{}^*, L_2{}^*, \ldots, L_n{}^*, \sigma_i{}^* = (X^*, W^*), Z^* \rangle$, where $X^* = g^b$, $W^* = \dfrac{g^{k_i \cdot (as \cdot h' + r')}}{P_1{}^{r'}}$, $P_1{}^* = g^c$, $Z^* = H_3(e(PK^{r'}, P_1)) \oplus (id_i || M_\beta)$, and returns SC^* to A.

Phase 2: A makes multiple queries as those in Phase 1. Note that A cannot ask $sk_{id_{l^*}}$ of n target receivers id_{l^*}, where $l = \{1, 2, \ldots, n\}$ during the key generation query, or SC^* during Receiver-Decrypt-and-Verify query.

Guess: In the end, A outputs its guess $\beta' \in \{0, 1\}$. If $\beta' = \beta$, C selects (V, ρ) from $H_3 - list$ and outputs ρ as the solution of BCDH problem.

Analysis: In User-sign and User-encrypt quires, since $X = \dfrac{g^{r'}}{g^{as \cdot h}} = g^{(r' - as \cdot h)}$, there is $r = r' - as \cdot h$, and $W = (\dfrac{g^{k_i}}{P_1})^{r'} \cdot P_1{}^{as \cdot h} = (\dfrac{g^{k_i}}{P_1})^{(r' - as \cdot h)} \cdot (\dfrac{g^{k_i}}{P_1})^{as \cdot h} \cdot P_1{}^{as \cdot h} = (\dfrac{g^{k_i}}{P_1})^{(r' - as \cdot h)} \cdot g^{k_i \cdot as \cdot h} = (\dfrac{g^{k_i}}{P_1})^r \cdot sk_{id_i}{}^h = sk_{id_i}{}^h pk_{id_i}{}^r$. Because $y_l = X^{(k_l - t)} = g^{r \cdot (k_l - t)} = \left(\dfrac{g^{k_l}}{g^t}\right)^r = \left(\dfrac{g^{k_l}}{P_1}\right)^r = pk_{id_l}{}^r$, where $l = \{1, 2, \ldots, n\}$, L_l can be calculated and the target signcryption can be realized.

During the challenge process, C sets $X^* = g^b$ and $P_1{}^* = g^c$. After knowing $pk_{id_{l^*}} = \dfrac{g^{k_{l^*}}}{P_1}$, C can get $y_{l^*} = X^{*(k_{l^*} - c)} = g^{b(k_{l^*} - c)} = (\dfrac{g^{k_{l^*}}}{g^c})^b = (\dfrac{g^{k_{l^*}}}{P_1})^b = pk_{id_{l^*}}{}^b$, and then get L_{l^*} by Lagrange interpolation function. Therefore, SC^* is the same as described in the actual attack process. If A's guess is correct, A needs to ask the random oracle function H_3 to get $V = e(PK^r, P_1) = e(g^{as \cdot b}, P_1) = e(g^{as \cdot b}, g^c) = e(g, g)^{abc \cdot s}$, Therefore, we can get the solution of BCDH problem $e(g, g)^{abc} = V \cdot e(g, g)^{s^{-1}}$.

In the attack phase, A performs $Q_{RD\&V}$ receiver-decrypt-and-verify quires. C selects V randomly from $H_3 - list$ to calculate $e(g, g)^{abc} = V \cdot e(g, g)^{s^{-1}}$ as the result of BCDH problem. Therefore, C can get the advantage $\varepsilon' \geq \varepsilon - \frac{Q_{H_3} Q_{RD\&V}}{2^k}$, and $\tau' \approx \tau + (2Q_{RD\&V} + Q_{UE})O(\tau_1)$, where τ_1 is the running time of e. From the above proof and BCDH problem, we can see that this scheme satisfies the confidentiality of signcryption.

5.2 Performance Evaluation

In this paper, we mainly evaluate the performance from signcryption efficiency and signcryption attributes.

In order to explore the signcryption efficiency, we mainly analyze its computing cost and communication traffic (i.e., length of signcryption). Tab. 1 shows the comparison results of the signcryption efficiency between the proposed scheme and prvious schemes.

Table 1: Comparison of the Signcryption Efficiency

Schemes	Mul	Exp	Log	Pair	Hash	Num	Length of Signcryption
Reference [22]	4	1	1	1	3	n+9	$3\|G_1\| + n\|ID\| + \|M\|$
Reference [23]	3	2n+2	1	1	2	9	$(n+3)\|G_1\| + n\|ID\| + \|M\|$
Reference [24]	5	1	1	3	3	n+11	$5\|G_1\| + 2n\|ID\| + \|M\|$
This scheme	1	n+3	1	1	3	11	$(n+2)\|G_1\| + \|ID\| + \|M\|$

Mul represents multiplication operation in G_1; *Exp* represents exponential operation in G_1; *Log* represents logical operation; *Pair* represents bilinear operation in G_2; *Hash* represents the hash operation in the signature and encryption step; *Num* represents the number of parameters; $|G_1|$ represents the length of elements in G_1; $|ID|$ represents the length of identity information; $|M|$ represents the length of plaintext message; n represents the number of receivers.

Tab. 2 shows the comparison results of signcryption attributes between the proposed scheme and previous schemes.

Table 2: Comparison of the Signcryption Attributes

Schemes	Unforgeability	Confidentiality	Model	Multiple Receivers	Anonymity of Receivers	Multiple Authorities	Resisting Collusion Attacks
Reference [22]	Y	Y	Random Oracle	Y	N	N	N
Reference [23]	Y	Y	Standard Model	Y	N	N	N
Reference [24]	Y	Y	Random Oracle	Y	Y	N	N
This scheme	Y	Y	Random Oracle	Y	Y	Y	Y

Compared with previous schemes, the proposed scheme has less $|ID|$ length and relatively moderate communication traffic in terms of signcryption efficiency. In order to ensure that the identities of receivers are not exposed in the signcrypted message, our scheme uses Lagrange interpolation to realize the anonymity of receivers. Lagrange interpolation involves many multiplications and exponential operations, so it increases the computing cost and affects the efficiency. However, the Lagrange formula can be calculated before the signcryption, so the operation in the signcryption step can be greatly reduced.

In terms of signcryption attributes, the signcryption scheme proposed in this paper satisfies unforgeability and confidentiality under a random oracle model. Compared with other schemes, the proposed scheme is more suitable for multiple receivers and can guarantee the anonymity of receivers. Importantly, the distributed key generation is realized by multiple authorities and can resist collusion attacks.

6 Conclusion

In order to allow patients to control their own EHRs initiative and share EHRs safely in blockchain, in this paper, we introduced multiple authorities into the identity-based signcryption scheme, and constructed a detailed MA-IBSC scheme for multiple receivers. The MA-IBSC scheme can not only resist the collusion attack of at most N-1 corrupted authorities, but also share the same signcryption message with multiple designated receivers. At the same time, the identity information of these receivers is anonymous. Under the assumptions of CDH and BCDH, it is proved that the proposed scheme is secure, that is, it satisfies unforgeability and confidentiality of signcryption.

Funding Statement: This work was supported by the National Key Research and Development Project of China (Grant No. 2017YFB0802302), the Science and Technology Support Project of Sichuan Province (Grant Nos. 2016FZ0112, 2017GZ0314, and 2018GZ0204), the Academic and Technical Leaders Training Funding Support Projects of Sichuan Province (Grant No. 2016120080102643), the Application Foundation Project of Sichuan Province (Grant No. 2017JY0168), and the Science and Technology Project of Chengdu (Grant Nos. 2017-RK00-00103-ZF, and 2016-HM01-00217-SF).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] L. Z. Xiong and Y. Q. Shi, "On the privacy-preserving outsourcing scheme of reversible data hiding over encrypted image data in cloud computing," *Computers, Materials & Continua*, vol. 55, no. 3, pp. 523–539, 2018.

[2] U. Premarathne, A. Abuadbba, L. Khalil, Z. Tari and A. Zomaya, "Hybrid cryptographic access control for cloud-based EHR systems," *IEEE Cloud Computing*, vol. 3, no. 4, pp. 58–64, 2016.

[3] G. Ramu, "A secure cloud framework to share EHRs using modified CP-ABE and the attribute bloom filter," *Education and Information Technologies*, vol. 23, no. 5, pp. 2213–2233, 2018.

[4] Z. Deng, Y. Ren, Y. Liu, X. Yin, Z. Shen *et al.,* "Blockchain-based trusted electronic records preservation in cloud storage," *Computers, Materials & Continua*, vol. 58, no. 1, pp. 135–151, 2019.

[5] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: https://nakamotoinstitute.org/bitcoin.

[6] P. B. Nichol, "Blockchain applications for healthcare: Blockchain opportunities are changing healthcare globally-innovative leaders see the change," 2016. [Online]. Available: http://www.cio.com/article/3042603/innovation/blockchain-applications/for-healthcare.html.

[7] M. Crosby and V. Kalyanaraman, *Blockchain Technology: Beyond Bitcoin*. Berkeley, USA: Applied Innovation Review, issue no. 2, 2015.

[8] A. Roehrs, C. A. da Costa and R. da Rosa Righi , "OmniPHR: A distributed architecture model to integrate personal health records," *Journal of Biomedical Informatics*, vol. 71, pp. 70–81, 2017.

[9] W. J. Gordon and C. Catalini, "Blockchain technology for healthcare: Facilitating the transition to patient-driven interoperability," *Computational and Structural Biotechnology Journal*, vol. 16, pp. 224–230, 2018.

[10] A. A. Omar, S. Rahman, A. Basu and S. Kiyomoto, "MediBchain: A blockchain based privacy preserving platform for healthcare data," in *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, Cham, Switzerland: Springer, pp. 534–543, 2017.

[11] S. Badr, I. Gomaa and E. Abd-Elrahman, "Multi-tier blockchain framework for IoT-EHRs system," *Procedia Computer Science*, vol. 141, pp. 159–166, 2018.

[12] L. X. Chen, W. K. Lee, C. C. Chang, K. K. R. Choo and N. Zhang, "Blockchain based searchable encryption for electronic health record sharing," *Future Generation Computer Systems*, vol. 95, pp. 420–429, 2019.

[13] X. Jiang, M. Z. Liu, C. Yang, Y. H. Liu and R. L. Wang, "A blockchain-based authentication protocol for WLAN mesh security access," *Computers, Materials & Continua*, vol. 58, no. 1, pp. 45–59, 2019.

[14] F. Tang, S. Ma, Y. Xiang and C. L. Lin, "An efficient authentication scheme for blockchain-based electronic health records," *IEEE Access*, vol. 7, pp. 41678–41689, 2019.

[15] R. Guo, H. Shi, Q. Zhao and D. Zheng, "Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems," *IEEE Access*, vol. 6, pp. 11676–11686, 2018.

[16] Y. Sun, R. Zhang, X. Wang, K. Gao and L. Liu, "A decentralizing attribute-based signature for healthcare blockchain," in *27th International Conference on Computer Communication and Networks (ICCCN)*, Hangzhou, China, pp. 1–9, 2018.

[17] Y. Zheng, "Digital signcryption or how to achieve cost (signature & encryption)," in *CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, B. S. Kaliski, CA, USA, Berlin, Heidelberg: Springer-Verlag, pp. 165–179, 1997.

[18] J. Malone-Lee, "Identity-based signcryption," in *Cryptology ePrint Archive*, Report 2002/098, 2002. [Online]. Available: http://eprint.iacr.org/2002/098.pdf.

[19] A. Karati and G. P. Biswas, "A practical identity based signcryption scheme from bilinear pairing," in *International Conference on Advances in Computing, Communications and Informatics*, Jaipur, India, pp. 832–836, 2016.

[20] C. X. Zhou, Y. Zhang and L. M. Wang, "A provable secure identity-based generalized proxy signcryption scheme," *International Journal of Network Security*, vol. 20, no. 6, pp. 1183–1193, 2018.

[21] S. Duan and Z. Cao, "Efficient and provably secure multi receiver identity based signcryption," in *Australasian Conference on Information Security & Privacy*, Australia, pp. 195–206, 2006.

[22] S. S. D. Selvi, S. S. Vivek and R. Srinivasan, "An efficient identity-based signcryption scheme for multiple receivers," in *Proceedings of the 4th International Workshop on Security: Advances in Information and Computer Security*, Berlin: Springer, pp. 71–88, 2009.

[23] B. Zhang and Q. L. Xu, "An ID-based anonymous signcryption scheme for multiple receivers secure in the standard model," in *Proceedings of the 2010 International Conference on Advances in Computer Science and Information Technology*, Berlin: Springer, vol. 20, pp. 15–27, 2010.

[24] X. Wang, J. Shu, W. Zheng, L. L. Liu and X. Fan, "New multi-receiver ID-based ring signcryption scheme," in *Unifying Electrical Engineering and Electronics Engineering*. X. Song, VA, New York, USA: Springer, pp. 2251–2257, 2014.