Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

An Efficient Authentication Scheme for Blockchain-Based Electronic Health Records

FEI TANG[1,2], SHUAI MA[1], YONG XIANG[3], (SENIOR MEMBER, IEEE), AND CHANGLU LIN[4]

[1]College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[2]School of Cyber Security and Information Law, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[3]School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia
[4]Key Laboratory of Network Security and Cryptography, Fujian Normal University, Fujian 350007, China

Corresponding author: Fei Tang (e-mail: tangfei@cqupt.edu.cn)

ABSTRACT In traditional Electronic Health Records (EHRs), medical-related information is generally separately controlled by different hospitals and thus it leads to inconvenience of information sharing. Cloud-based EHRs solve the problem of information sharing in the traditional EHRs. However, cloud-based EHRs suffer centralized problem, i.e., cloud service center and key-generation center. This paper works on creating a new EHRs paradigm which can help in dealing with the centralized problem of cloud-based EHRs. Our solution is to make use of the emerging technology of blockchain to EHRs (denoted as blockchain-based EHRs for convenience). Firstly, we formally define the system model of blockchain-based EHRs in the setting of consortium blockchain. In addition, authentication issue is very important for EHRs. However, existing authentication schemes for blockchain-based EHRs have their own weak points. Therefore, in this work, we also propose an authentication scheme for blockchain-based EHRs. Our proposal is an identity-based signature scheme with multiple authorities which can resist collusion attack out of N from $N-1$ authorities. Furthermore, our scheme is provably secure in the random oracle model and has more efficient signing and verification algorithms than existing authentication schemes of blockchain-based EHRs.

INDEX TERMS Electronic health records, blockchain, identity-based signatures, multiple authorities.

I. INTRODUCTION

Traditional paper-based health records apparently are inconvenient for information interchange or sharing. The technology of Electronic Health Records (EHRs) [7], [8], [19], [20] provides a novel way to collect and manage health-related information. EHRs are an information system which maintains medical records in the process of patients' treatment or health management. It contains various kinds of health information and realizes the summary or integration of different electronic medical information and satisfy the management needs of hospitals and related research institutions. EHRs are more convenient than traditional paper-based health records for information storage and retrieval. In EHRs, all medical-related data are digitized and stored in the server of hospital. Then, when a patient goes back to the hospital, he or the hospital can search previous information, including names of the patient and doctor, time, diagnosis, and so on. As an important application in the medical field, EHRs have attracted wide attention. Many standards have been proposed [19], [20] for EHRs. In addition, many papers considered the security and privacy issues in EHRs systems [7], [8], [19].

However, there exists many problems in traditional EHRs. First of all, generally, medical-related data are independently stored in different hospitals or research institutions since they have their own independent database. Therefore, when a patient transfers from a hospital to another one, he needs to obtain medical examinations once again. This obviously will lead to waste of medical information resources and increase patients' body and financial burdens. Secondly, in EHRs systems, only the authorities, such as hospitals, have data. Hence, if there is a dispute between hospital and patient, then the hospital will always win since it can tamper the medical records or even delete them. It is not fair for patients.

In order to solve the problem of information sharing in the traditional EHRs, researchers introduced the notion of could-based EHRs [11]–[14], [16], [21], [25]. The cloud-based EHRs can be seem as an application of the could computing technology in EHRs. In cloud-based EHRs systems, there

still needs a cloud service provider who plays the role of authority. As shown in Figure 1, all medical-related data, from doctor, pharmacy, diagnostic laboratory, insurance center, and so on, will be uploaded to the cloud server. Then, users can search and download useful information from the could server. If several organizations share a same cloud server, then they can share the data with a convenient way. Next, when patients transfer from a hospital to another one, the new hospital can obtain patients' medical-related data from the cloud and thus they have no need to, once again, get medical examinations. Therefore, cloud-based EHRs solve the problem of information sharing in the traditional EHRs. In addition, in cloud-based EHRs, all data are only maintained by the authority, i.e., cloud service provider, and thus the hospitals and other organizations could tamper the medical-related data only when they collude with the authority.

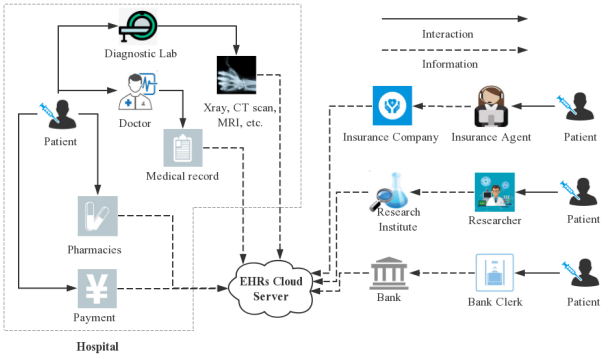

Figure 1. Cloud-based EHRs.

Cloud-based EHRs solve the problem of information sharing, and make hospitals and other organizations cannot tamper the data by themselves. However, there also has some problems in the cloud-based EHRs. Firstly, if there has dispute between hospital and patient, then the hospital can collude with the cloud service provider to tamper or even delete the data. Therefore, as in many other kinds of cloud-based systems, we need to put our trust on the cloud server. Whereas, if the cloud server is attacked or it is malicious, then patients' privacy is a big problem. Secondly, in identity-based and attribute-based cryptosystems for cloud-based EHRs, e.g., [12], [21], [25], there is a key generation center (KGC) who is responsible for key generation for all users. Actually, it is well known that KGC knows all users' secret keys.

This paper works on creating a new EHRs paradigm which can help in dealing with the problems in cloud-based EHRs. Our solution is to make use of the emerging technology of blockchain which is derived from Bitcoin [15]. Generally speaking, blockchain can be seem as a decentralized and distributed database. There is authority in traditional network architectures or application systems, such as KGC, cloud service provider, and so on. The decentralized feature of blockchain gets rid of such dependence on authority. Therefore, many people considered the applications of blockchain in different types of real-world scenarios, including EHRs, we call it blockchain-based EHRs. For example, the works of [3], [4], [24] designed a broad framework for blockchain-based EHRs. Zhang et al. [23] and Omar et al. [27] made use of encryption technology to protect the confidentiality of the medical records. Xu et al. [22] focus on the privacy issue of EHRs and designed a new framework based on blockchain and homomorphic encryption.

Authentication is very important for blockchain-based EHRs. It is different from the case of cryptocurrency which is anonymous and thus there is no authentication mechanism for users, the data in blockchain-based EHRs must be authenticated, such as diagnosis from doctors. However, there are few works about such issue. Sun et al. [26] designed a decentralized attribute-based signature scheme for blockchain-based EHRs. In their scheme, each authority agency is in charge of one or more attributes. That is to say, the different attributes of the user are issued by one or more authority agencies. Therefore, the scheme of [26] is vulnerable to the collusion attack of authorities. Guo et al. [6] also construed an attribute-based signature scheme with multiple authorities for blockchain-based EHRs. Their scheme can resist collusion attack out of N from $N-1$ corrupted authorities. However, in their scheme, each patient has a blockchain of healthcare alone which is incompatible with the property of blockchain.

A. OUR MOTIVATIONS AND CONTRIBUTIONS

There are few works considered the authentication issue for blockchain-based EHRs where the scheme of [26] suffers from the problem of collusion attack and the model of [6] is incompatible with blockchain. In addition, both of [26] and [6] have not thought about the roles of the organizations, such as hospitals, medical insurance companies, scientific research institutions, pharmaceutical companies, and so on, which is inappropriate in real-world application. Furthermore, the signing and verification costs of both schemes [6], [26] are high. However, as said before, the authentication issue is very important for blockchain-based EHRs. Therefore, in this work, we further consider this problem for blockchain-based EHRs. The main contributions of this work are as follows:

- Firstly, as the roles of organizations of EHRs have not been considered in the models of [6], [26], we re-define the model of blockchain-based EHRs. Our model is defined in the setting of consortium blockchain [9], [10], which corresponds to the real-world of EHRs.
- Secondly, the authentication efficiency is very important for blockchain-based EHRs, especially when the amount of data is large. Therefore, in this paper, we design an efficient authentication scheme for blockchain-based EHRs. Our proposal is an identity-based signature scheme with multiple authorities (MA-IBS) which has both efficient signing and verification algorithms and can resist collusion attack.

- Finally, we prove the security of the proposed scheme, in the random oracle model, under the computational Diffie-Hellman assumption. Meanwhile, we evaluate the performance of the scheme and compare it with the existing two authentication schemes [6], [26] in blockchain-based EHRs.

B. PAPER ORGANIZATION

The rest of this paper is organized as follows. Section II describes the formal definition of blockchain-based EHRs. In Section III, we present some preliminaries including definition of MA-IBS, security model, and the relationship between MA-IBS and blockchain-based EHRs. The proposed scheme is given in Section IV and security proof and performance evaluation are given in Section V. Finally, Section VI concludes this work.

II. MODEL OF BLOCKCHAIN-BASED EHR

We divide the users of blockchain-based EHRs into three levels. The first level, denoted by Level 0, is the EHRs server. The second level, denoted by Level 1, contains several kinds of organizations, such as hospitals, medical insurance companies, scientific research institutions, pharmaceutical companies, and so on. The third level, denoted by Level 2 which corresponds to the employees of the Level 1 users, consists of doctors, researchers, patients, insurance agents, and so on. In blockchain-based EHRs systems, all medical-related data will be distributed stored by all Level 1 users who can reach a consensus, for the authenticity of the shared data, based on a specific mechanism. The responsibility of Level 2 users is that generate medical-related information, such as medical records from doctors, insurance policies from insurance agent, and so on. The authenticity of such information can be guaranteed by a proper authorization mechanism from Level 1 users to their employees.

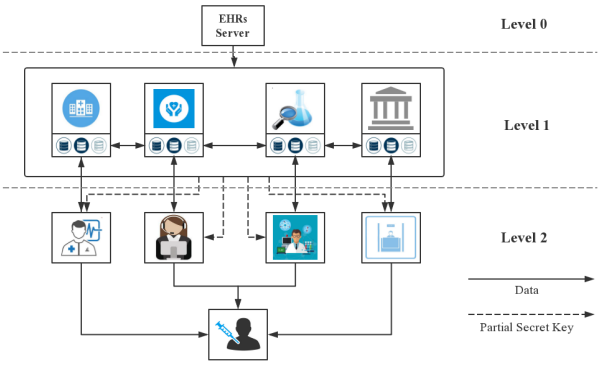

Figure 2. Blockchain-based EHRs.

We define the model, as shown in Figure 2, of blockchain-based EHRs in the setting of consortium blockchain.

- Level 0: is the EHRs server which is in charge of the generation of system parameters. EHRs server in blockchain-based EHRs is different from that in the cloud-based EHRs. In cloud-based EHRs, the medical-related data is stored only by the cloud service provider. However, in blockchain-based EHRs, the only responsibility of EHRs server is that choosing public parameters for all users. The existence of such a server is reasonable for blockchain systems, for example, Nakamoto [15] has chosen ECDSA and SHA-256 as the public parameters for Bitcoin system. Moreover, the KGC in identity-based and attribute-based cryptosystems of cloud-based EHRs [11], [12], knows all users' secret key. But the server in blockchain-based EHRs does not having the right to generate users' secret keys.
- Level 1: contains hospitals and other organizations. These users correspond to the authorities in consortium blockchain. All data are uploaded and stored by all authorities separately. In addition, Level 1 users also play the roles of key generation authorities and to generate Level 2 users' secret keys. However, for consortium blockchain of EHRs, we require that no one can control or obtain Level 2 users' secret keys. This is very important to understand the difference between cloud-based EHRs and blockchain-based EHRs.
- Level 2: is composed of doctors, researchers, insurance agents, and so on. These users correspond to the employees of Level 1 users, for example, doctor works for a hospital. The responsibility of Level 2 users is to provide specific medical-related information. For example, doctors give diagnosis, insurance agents sign insurance policy, and so on.

Authentication is one of the most important problems for EHRs because we need to ensure the authenticity of medical records in consortium blockchain. Corresponding to the system model, authentication of blockchain-based EHRs contains the following two cases:

- Case 1 (authentication of Level 2): It is means that the data given by Level 2 users need to authenticated by them. For example, we need to ensure the authenticity of a diagnosis from some doctor.
- Case 2 (authentication of Level 1): This case is corresponding to the authenticity of block data given by Level 1 users, i.e., authorities.

III. DEFINITIONS

In this section, we give several definitions for our work, including the definition of MA-IBS, security model of MA-IBS and relationship between MA-IBS and blockchain-based EHRs.

A. MA-IBS

The concept of identity-based signature (IBS) was introduced by Shamir [18]. Generally, an IBS scheme consists of four algorithms: **System Setup** algorithm produces public parameters for the other three algorithms and master secret key; **Key Generation** algorithm, which is executed by a key generation center, produces users' signing key; **Sign** and

Verify algorithms are responsible for signing and verification of signatures, respectively.

As far as we know, there has no work considered IBS scheme in the setting of multiple authorities. However, in fact, it is very natural to define the notion of IBS with multiple authorities (MA-IBS). In MA-IBS scheme, there has one more algorithm, **Authority Setup**, to generate all authorities' master secret keys. To satisfy the requirements of blockchain-based EHRs, we further adjust the definition. Our MA-IBS for blockchain-based EHRs contains the following seven algorithms:

- **System Setup**: The system setup algorithm is run by EHRs server who takes as input a security parameter λ. Then it outputs system public parameters $params$.
- **Authority Setup**: The authority setup algorithm is interactively executed by all authorities who take as inputs public parameters $params$ and their identities $ID_1, ..., ID_N$. Then they output their master secret keys $SK_{ID_1}, ..., SK_{ID_N}$.
- **Key Generation**: The key generation algorithm is also interactively executed by all authorities who take as inputs the public parameters $params$, their master secret keys $SK_{ID_1}, ..., SK_{ID_N}$, and user's identity id_i. Then they output user id_i's secret key sk_{id_i}.
- **User-Sign**: This sign algorithm is run by signer id_i who takes as inputs public parameters $params$, secret key sk_{id_i} and message m. Then he outputs a signature σ_i.
- **User-Verify**: This verification algorithm can be publicly executed by all users who take as inputs the signer's identity id_i, message m, and signature σ_i. Then it outputs $Accept$ if it is valid; Else, outputs $Reject$.
- **Authority-Sign**: This sign algorithm is run by an authority ID_i who takes as inputs public parameters $params$, its master secret key SK_{ID_i} and message M. Then it outputs a signature δ_i.
- **Authority-Verify**: This verification algorithm can be publicly executed by anyone who takes as inputs identity ID_i, message M, and a signature δ_i. Then it outputs $Accept$ if it is valid; Else, outputs $Reject$.

B. SECURITY MODEL OF MA-IBS

The security model of the adapted MA-IBS is defined by the following game.

- **System Setup**: Challenger $\mathcal{C}$ chooses a security parameter λ and runs the system setup algorithm to generate system public parameters $params$. Then, $\mathcal{C}$ gives $params$ to adversary $\mathcal{A}$.
- **Authority Setup**: Challenger $\mathcal{C}$ runs the authority setup algorithm to generate master secret keys $SK_{ID_1}, ..., SK_{ID_N}$ for authorities whose identities are $ID_1, ..., ID_N$, respectively.
- **Queries**: Adversary $\mathcal{A}$ can make the following four kinds of queries to $\mathcal{C}$:
  - Master secret key queries: $\mathcal{A}$ issues a request for some authorities $ID_{i\in Q_M} \subset \{ID_1, ..., ID_N\}$ for their master secret key, where Q_M denotes the index set of the identities of corrupted authorities. For such a request, $\mathcal{C}$ transmits $SK_{ID_{i\in Q_M}}$ to $\mathcal{A}$.
  - Key generation queries: upon receiving an identity id_i, $\mathcal{C}$ then returns back a corresponding secret key sk_{id_i} to $\mathcal{A}$. Let Q_K be the set of identities which were given to the key generation queries.
  - User-sign queries: upon receiving a message m_i and an identity id_i, $\mathcal{C}$ then returns back a corresponding signature σ_i to $\mathcal{A}$. Let Q_{US} be the set of (m_i, id_i) where were given to the user-sign queries.
  - Authority-sign queries: upon receiving a message M_i and an identity ID_i, $\mathcal{C}$ then returns back a corresponding signature δ_i to $\mathcal{A}$. Let Q_{AS} be the set of (M_i, ID_i) where were given to the authority-sign queries.
- **Forgery**: Finally, adversary $\mathcal{A}$ outputs a tuple of forgery (id^*, m^*, σ^*) or (ID^*, M^*, δ^*).

We say that the adversary $\mathcal{A}$ wins the game if one of the following two cases holds:

- Case 1: If the forgery is (id^*, m^*, σ^*), then **User-Verify**$(id^*, m^*, \sigma^*) = Accept \bigwedge Q_M \neq [1, N] \bigwedge id^* \notin Q_K \bigwedge (m^*, id^*) \notin Q_{US}$.
- Case 2: If the forgery is (ID^*, M^*, δ^*), then **Authority-Verify**$(ID^*, M^*, \delta^*) = Accept \bigwedge ID^* \notin Q_M \bigwedge (M^*, ID^*) \notin Q_{AS}$.

The advantage that $\mathcal{A}$ wins the above game is defined as $\mathbf{Adv}^{UF-CMA}_{MA-IBS,\mathcal{A}}(\lambda)$.

Definition 1: We say that an MA-IBS scheme is $(t, q_H, q_M, q_K, q_{US}, q_{AS}, \epsilon)$-unforgeable if, for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, it's advantage ϵ is negligible, where it runs the game at most t in time, and makes at most q_H hash function queries (in the random oracle model), q_M master secret key generation queries, q_K key generation queries, q_{US} user-signing queries, and q_{AS} authority-signing queries, respectively, while the .

C. MA-IBS FOR BLOCKCHAIN-BASED EHR

We now describe the relationship between the concept of adapted MA-IBS and the model of blockchain-based EHRs described in Section II.

First, the Level 0 user, i.e., EHRs server, runs **System Setup** algorithm to produce the system public parameters. Then, Level 1 users, i.e., authorities, interactively execute the **Authority Setup** algorithm to generate their master secret keys. In this phase, we require that authorities can reach a consensus on the authenticity of all master secret keys, although there has no trusted center. Next, Level 2 users can obtain their signing key from Level 1 users who distributed execute the **Key Generation** algorithm. Then, Level 2 users will produce some medical-related information, e.g., diagnosis. To ensure the non-repudiation of such information, we require that Level 2 users run **User-Sign** algorithm to sign it. Finally, an authority collects the medical-related information and signatures from Level 2 users to form a block. To ensure

the authenticity of block, we require the authority signs it by running **Authority-Sign** algorithm. As shown in Figure 3, the message M taken as input to this signing algorithm consists of previous block hash, time stamp, Merkle root of medical records, and identity of the authority.

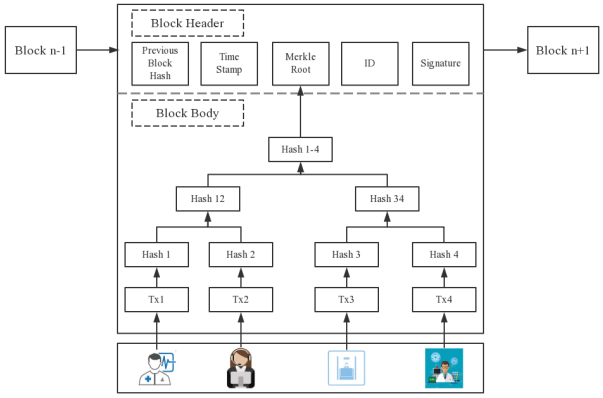

Figure 3. Block of Blockchain-based EHRs.

Note that there is a difference between blockchain-based EHRs and Bitcoin [15]. In Bitcoin system which is based on public blockchain, to reach an agreement on a block, it requires the node who uploads the block to find a random number to satisfy a specific requirement of hash function, i.e., proof of work. However, in the blockchain-based EHRs which are based on consortium blockchain, authorities sign the block, and thus the authenticity of the block is always guaranteed. Therefore, in Bitcoin system, if someone wants to change or delete a block, it only needs to change or delete the block in over 51% nodes' blockchain, whereas it needs to change or delete the block in all authorities' blockchain in blockchain-based EHRs systems.

IV. THE PROPOSED SCHEME

We now construct an efficient MA-IBS scheme without having a trusted authority for blockchain-based EHRs systems. In the beginning, we give the algebraic tool and complexity assumption used in our scheme.

A. BILINEAR MAP AND COMPLEXITY ASSUMPTION

In the construction of our MA-IBS scheme, we will make use of bilinear map as the basic tool. Therefore, we briefly introduce the concept of bilinear map.

Let $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic multiplicative groups, where $\mathbb{G}$ is generated by an element g, i.e, $\mathbb{G} =< g >$. Groups $\mathbb{G}$ and $\mathbb{G}_T$ have the same big prime order p. We say that $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an admissible bilinear map if it satisfies the following three properties:

1) Bilinearity: for all $a, b \in \mathbb{Z}_p, e(g^a, g^b) = e(g, g)^{a \cdot b}$.
2) Non-degeneracy: there exists $g^c, g^d \in \mathbb{G}$, for $c, d \in \mathbb{Z}_p$, such that $e(g^c, g^b) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is the identity element of group $\mathbb{G}_T$.
3) Computability: there is an efficient algorithm to compute $e(g^a, g^b) \in \mathbb{G}_T$ for all $a, b \in \mathbb{Z}_p$.

The security of our MA-IBS scheme is based on the complexity assumption of Computational Diffie-Hellman (CDH) problem which means that given $g, g^a, g^b \in \mathbb{G}$, where a and b are randomly and independently chosen from $\mathbb{Z}_p^*$, there has no PPT algorithm can compute $g^{ab} \in \mathbb{G}$.

B. CONSTRUCTION

In our MA-IBS scheme, N authorities have no need to be honest. Therefore, it satisfies the requirements of blockchain-based EHRs. In fact, the scheme can tolerate at most $N-1$ corrupted authorities to launch collusion attack.

- **System Setup:** EHRs server takes as input a security parameter λ to establish system public parameters for all users. First of all, it chooses two multiplicative cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ with a big prime order p, and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let g be a generator of the group $\mathbb{G}$. Next, it chooses two cryptographic hash functions $H : \{0,1\}^* \to \mathbb{G}$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$, respectively. System parameters of the EHRs system are $params = \{\mathbb{G}, \mathbb{G}_T, p, g, e, H, H_1, N\}$, where N is the number of authorities.
- **Authority Setup:** In this algorithm, N authorities establish their master secret keys, $SK_{ID_1}, ..., SK_{ID_N}$. It consists of the following three phases:
  - Phase 1 (generation of parameter $h \in \mathbb{G}$): All authorities are working from the same system parameters $params$ and collaborating together to generate a verification parameter $h \in \mathbb{G}$.
    1) Each authority ID_i chooses a random $(N-1)$-degree polynomial $\mathcal{H}_i(z)$ over $\mathbb{Z}_p^*$:
    $$\mathcal{H}_i(z) = c_{i0} + c_{i1}z + \cdots + c_{i(N-1)}z^{N-1}.$$
    Then, it computes and broadcasts $C_{ik} = g^{c_{ik}} \pmod p$ for $k = 0, 1, ..., N-1$. Next, it computes some secret values $t_{ij} = \mathcal{H}_i(H_1(ID_j)) \pmod p$ for $j = 1, ..., N$. Finally, it sends t_{ij} to ID_j for $j \neq i$.
    2) Each authority ID_i verifies the equation $g^{t_{ji}} \stackrel{?}{=} \prod_{k=0}^{N-1}(C_{jk})^{H_1(ID_i)^k}$ holds or not. If it holds, then ID_j is considered to be honest. Otherwise, authority ID_j will receive a complaint from ID_i. Then, ID_j needs to broadcast values t_{ji} so that it passes the verification.
    3) After the above interactions, a random parameter h can be generated as $h = \prod_{i=1}^{N} C_{i0} \pmod p$. Note that $h = g^{c_{10}+\cdots+c_{N0}}$ and the logarithm of g for h is unknown to everyone.
  - Phase 2 (generation of master secret key): This phase contains the following steps.
    1) Each authority ID_i, for $i = 1, ..., N$, randomly chooses two $(N-1)$-degree polynomials on $\mathbb{Z}_p^*$:
    $$\mathcal{F}_i(x) = a_{i0} + a_{i1}x + \cdots + a_{i(N-1)}x^{N-1},$$

$$\mathcal{F}_i'(x) = b_{i0} + b_{i1}x + \cdots + b_{i(N-1)}x^{N-1}.$$

Then, it computes and broadcasts $B_{ik} = g^{a_{ik}}h^{b_{ik}}$, for $k = 0, 1, ..., N-1$. In addition, it also computes secret values $s_{ij} = \mathcal{F}_i(H_1(ID_j)) \pmod p$ and $s'_{ij} = \mathcal{F}'_i(H_1(ID_j)) \pmod p$, for $j = 1, ..., N$. Finally, it sends s_{ij} and s'_{ij} to ID_j for $j \neq i$.

2) Each authority ID_i checks the equation $g^{s_{ji}}h^{s'_{ji}} \stackrel{?}{=} \prod_{k=0}^{N-1}(B_{jk})^{H_1(ID_i)^k} \pmod p$ holds or not. If it holds, the secret sharing from ID_j is valid; Otherwise, ID_i broadcasts a complaint against ID_j.
3) If authority ID_j is complained, then it needs to broadcast values (s_{ij}, s'_{ij}) that satisfy equation. If the disclosed (s_{ij}, s'_{ij}) still does not match, ID_j has to keep proving itself to be honest until the equation is true.
4) Note that the master secret key that interactively established by N authorities is $s = \sum_{i=1}^{N} a_{i0}$. If there has less than N authorities are corrupted, then they cannot recover the value s. The master secret key of authority ID_i is

$$SK_{ID_i} = a_{i0}.$$

– Phase 3 (generation of master public key): According to the above two phases, each authority has broadcasted values $\{A_{i0} = g^{a_{i0}}\}_{i\in[1,N]}$ which can be verified publicly. Therefore, the master public key can be computed as

$$y = \prod_{i=1}^{N} A_{i0} = \prod_{i=1}^{N} g^{a_{i0}} = g^{\sum_{i=1}^{N} a_{i0}} = g^s \in \mathbb{G}.$$

After the above three phases, each authority adds parameters y and $\{(ID_i, A_{i0})\}_{i=1}^{N}$ to $params$.

$$params := \{\mathbb{G}, \mathbb{G}_T, p, g, y, e, H, H_1, \{(ID_i, A_{i0})\}_{i=1}^{N}\}.$$

- **Key Generation:** When a user registers to the EHRs system, it can obtain a secret key sk_{id_i} from authorities.
  – Phase 1 (generation of partial secret key): Each authority ID_j computes a value $psk_{id_i,j} = H(0||id_i)^{a_{j0}}$ and secretly transmits it to id_i.
  – Phase 2 (verification of partial secret key): After receiving the partial secret key $psk_{id_i,j}$ from authority ID_j, user id_i can verify its validity by checking the equation $e(psk_{id_i,j}, g) \stackrel{?}{=} e(H(0||id_i), A_{j0})$ holds or not. If it holds, then the partial secret key is correct. Otherwise, the authority ID_j needs to retransmit the value that satisfies the equation.
  – Phase 3 (generation of secret key): After receiving all partial secret keys. User id_i computes his secret key as

$$sk_{id_i} = \prod_{j=1}^{N} psk_{id_i,j} = \prod_{j=1}^{N} H(0||id_i)^{a_{j0}} = H(0||id_i)^s.$$

- **User-Sign:** To sign a message $m \in \{0,1\}^*$, user id_i does the following three phases:
  – Phase 1: randomly choose an integer $r \in \mathbb{Z}_p^*$ and computes $u = H(0||id_i)^r \in \mathbb{G}$.
  – Phase 2: compute $t = H_1(m||u) \in \mathbb{Z}_p^*$.
  – Phase 3: compute $v = sk_{id_i}^{r+t} \in \mathbb{G}$.

  The signature on message m is $\sigma = (u, v)$.
- **User-Verify:** One can verify the validity of a signature $\sigma = (u, v)$ on a message m from signer id_i.
  – Phase 1: compute $t = H_1(m||u) \in \mathbb{Z}_p^*$.
  – Phase 2: check the following equation holds or not

$$e(v, g) \stackrel{?}{=} e(u, y) \cdot e(H(0||id_i)^t, y).$$

  If it holds, accept the signature; Else reject it.
- **Authority-Sign:** Before upload a block, denoted by M, to the chain, authority ID_i needs to sign it as follows.
  – Compute $\delta = H(1||M)^{a_{i0}} \in \mathbb{G}$.

  The signature on block data M is δ.
- **Authority-Verify:** Anyone can check the validity of a signature δ on block data M from an authority ID_i.
  – Check the following equation holds or not

$$e(\delta, g) \stackrel{?}{=} e(H(1||M), A_{i0}).$$

  If it holds, accept the signature; Else reject it.

C. CORRECTNESS

1) Correctness of Users' Signatures

The correctness of the users' signatures can be easily verified by the following equation:

$$\begin{aligned} e(v, g) &= e(sk_{id_i}^{r+t}, g) \\ &= e(H(0||id_i)^{s\cdot(r+t)}, g) \\ &= e(H(0||id_i)^{(r+t)}, g^s) \\ &= e(H(0||id_i)^r, y) \cdot e(H(id_i)^t, y) \\ &= e(u, y) \cdot e(H(0||id_i)^t, y). \end{aligned}$$

2) Correctness of Authorities' Signatures

The correctness of the authorities' signatures can be easily verified by the following equation:

$$\begin{aligned} e(\delta, g) &= e(H(1||M)^{a_{i0}}, g) \\ &= e(H(1||M), g^{a_{i0}}) \\ &= e(H(1||M), A_{i0}). \end{aligned}$$

D. BATCH VERIFICATION

Based on the EHRs system architecture as presented in Section II, once a user receives a health related message from another user, for example, patient receives diagnostic report from a doctor, he needs to verify the signature to ensure the validity of the message. In addition, when authorities upload EHRs information into the blockchain, they also need to verify the validity of all messages. We can make use of the technique of batch verification [1] to improve the efficiency of verification.

1) Verify n Signatures from a Same Signer

In some cases in EHRs system, we need to verify many signatures from a same signer at once. For example, system server wants to verify the validity of insurance policies from an agent during some time range.

Given n signatures on n messages, $(m_j, \sigma_j = (u_j, v_j))$, for $j = 1, ..., n$, which were signed by a same signer id_i, to verify their validity, we only need to check that the equation $e(\prod_{j=1}^{n} v_j, g) \stackrel{?}{=} e(\prod_{j=1}^{n} u_j, y) \cdot e(H(0||id_i)^{\sum_{j=1}^{n} t_j}, y)$, where $t_j = H_1(m_j||u_j)$, holds or not. Correctness of the batch verification is as follows:

$$\begin{aligned} & e(\prod_{j=1}^{n} v_j, g) \\ = \; & e(\prod_{j=1}^{n} sk_{id_i}^{r_j+t_j}, g) \\ = \; & e(H(0||id_i)^{s\cdot\sum_{j=1}^{n}(r_j+t_j)}, g) \\ = \; & e(H(0||id_i)^{\sum_{j=1}^{n}(r_j+t_j)}, g^s) \\ = \; & e(H(0||id_i)^{\sum_{j=1}^{n} r_j}, y) \cdot e(H(0||id_i)^{\sum_{j=1}^{n} t_j}, y) \\ = \; & e(\prod_{j=1}^{n} u_j, y) \cdot e(H(id_i)^{\sum_{j=1}^{n} t_j}, y). \end{aligned}$$

2) Verify n Signatures from n Signers

In some other cases of EHRs system, we need to verify many signatures from many different signer at once. For example, some patient wants to verify the validity of diagnostic reports from n different doctors.

Given n signatures $(\sigma_j = (u_j, v_j)$ on n messages m_j, for $j = 1, ..., n$, from n signers $id_1, ..., id_n$, respectively. These signatures are valid if and only if $e(\prod_{j=1}^{n} v_j, g) = e(\prod_{j=1}^{n} u_j, y) \cdot e(\prod_{j=1}^{n} H(id_j)^{t_j}, y)$, where $t_j = H_1(m_j||u_j)$. Correctness of the batch verification is as follows:

$$\begin{aligned} & e(\prod_{j=1}^{n} v_j, g) \\ = \; & e(\prod_{j=1}^{n} sk_{id_j}^{r_j+t_j}, g) \\ = \; & e(\prod_{j=1}^{n} H(0||id_j)^{s\cdot(r_j+t_j)}, g) \\ = \; & e(\prod_{j=1}^{n} H(0||id_j)^{(r_j+t_j)}, g^s) \\ = \; & e(\prod_{j=1}^{n} H(0||id_j)^{r_j}, y) \cdot e(\prod_{j=1}^{n} H(0||id_j)^{t_j}, y) \\ = \; & e(\prod_{j=1}^{n} u_j, y) \cdot e(\prod_{j=1}^{n} H(0||id_j)^{t_j}, y). \end{aligned}$$

V. SECURITY AND PERFORMANCE

In this section, we prove the security of our MA-IBS scheme and evaluate its performance.

A. SECURITY PROOF

The proof is given in the random oracle model. As said in Subsection III-B, adversary's forgery has two possible cases: **User-Sign** signatures and **Authority-Sign** signatures. Challenger $\mathcal{C}$ needs to have different strategies to interact with the adversary. $\mathcal{C}$ cannot know the adversary's choice until the game ends. Therefore, it randomly guesses adversary's choice in the beginning with a probability of 1/2.

1) Case of User-Sign Signature Forgery

For the case of **User-Sign** signature forgery, the core technique of our proposal is that taking distributed key generation technique [5] to a centralized IBS scheme [2]. It seems that the security of our scheme directly holds based on the securities of the two schemes. However, this statement is not true. In the scheme of [2] which is based on the assumption of CDH problem, the key of security proof is that the challenger sets one element g^a of the CDH instance as the master public key $y := g^a$. Then, it sets the another element g^b of the CDH instance as $H(id_{i^*}) := g^b$. Finally, if the adversary chooses id_{i^*} as his challenge identity and outputs a valid forgery, then the challenger can rewind the tape of random oracle H and recover g^{ab} which means that solving the CDH problem. However, in our proposed scheme, the master public key is $y := g^s$, where s is randomly generated by N authorities and no one know it. Hence, in security proof, the challenger cannot set the master public key as $y := g^a$, and thus cannot take advantage of the adversary's forgery to compute g^{ab}.

To resolve the dilemma, we make use of the approach of hybrid proof. We first define three games as follows:

- Game $\mathcal{G}_0$: This game corresponds to the honest execution of the security game defined in Definition 1.
- Game $\mathcal{G}_1$: In this game, we set the master secret key as $y := g^{as}$ where a is the exponent of the CDH instance and s is the master secret key randomly generated by all authorities, respectively. No one knows a and s.
- Game $\mathcal{G}_2$: In this game, the master secret key also is $y := g^{as}$ where a is the exponent of the CDH instance. However, it is different than that in $\mathcal{G}_1$, in this game, the challenger plays the role of all authorities and thus it knows the value s.

Then we prove that the advantages of any PPT adversary to attack our scheme in three games are identical and its advantage in $\mathcal{G}_2$ is negligible. We have the following three lemmas:

Lemma 1: There has no PPT adversary can distinguish $\mathcal{G}_0$ and $\mathcal{G}_1$ if the distributed key generation technique is secure.

Lemma 2: There has no PPT adversary can distinguish $\mathcal{G}_1$ and $\mathcal{G}_2$.

Lemma 3: The advantage of any PPT adversary $\mathcal{A}$ in game $\mathcal{G}_2$ is negligible.

The proofs of the above three lemmas are presented in Appendix. Finally, we can easily observe that the advantage of the adversary $\mathcal{A}$ in game $\mathcal{G}_0$ is also negligible which is our expected result.

Theorem 1: The MA-IBS scheme for blockchain-based EHRs is $(t, q_H, q_{H_1}, q_M, q_K, q_{US}, q_{AS}, \epsilon)$-unforgeable, for the case of **User-Sign** signature forgery in the random oracle model, assuming the CDH problem is hard.

The proof of this theorem is presented in Appendix.

2) Case of Authority-Sign Signature Forgery

If the adversary's forgery is the case of **Authority-Sign** signature, then the proof is simpler than the case of **User-Sign** signature. Firstly, challenger randomly guesses adversary's challenge point ID_{i^*} and sets its public key as g^a. Then, in the random oracle queries, challenger randomly chooses a point M_{i^*} as the adversary's challenge and set it

as g^b. Finally, if the adversary chooses ID_{i^*} and M_{i^*} as his challenge and outputs a valid forgery, then the challenger can rewind the tape of random oracle H and recover g^{ab} which means that solving the CDH problem.

Theorem 2: The MA-IBS scheme for blockchain-based EHRs is $(t, q_H, q_{H_1}, q_M, q_K, q_{US}, q_{AS}, \epsilon)$-unforgeable, for the case of **Authority-Sign** signature forgery in the random oracle model, assuming the CDH problem is hard.

The proof of this theorem is given in Appendix.

B. PERFORMANCE EVALUATION

We denote T_{par} as the time to perform paring operations, T_{mtp} as the map-to-point hash operations, T_{mul} as the multiplication operations in group $\mathbb{G}_T$, and T_{exp} as the exponentiation operations in group $\mathbb{G}$, respectively. Because these operations dominate the costs of signing and verification algorithms, we only consider these four operations and neglect the other operations such as hash function $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. Java Pairing-Based Cryptography Library (JPBC) is used to measure the run times of the above operations. We obtain the results: T_{par} is 5.796 ms, T_{mtp} is 1.293 ms, T_{mul} is 0.031 ms and T_{exp} is 5.786 ms within hardware platform of an Intel i7-8550U processor with 2.0 GHz clock frequency, 8 gigabytes memory and executed in Windows 10 operating system.

We compare our scheme to the only two exiting two authentication schemes [6], [26] for blockchain-based EHRs with respect to signing cost, verification cost, communication cost, and whether the scheme can resist collusion attack. The results are listed in Table 1, where t is the number of users attributes and N is the number of the authorities and we assume that $t = N = 5$ in both schemes schemes [26] and [6]. In addition, since the sizes of elements in the chosen groups $\mathbb{G}$ and $\mathbb{G}_T$ are 40 and 128 bytes, respectively. As shown in Table 1, our proposed authentication scheme for blockchain-based EHRs has lower computation and communication costs compared to the only two existing authentication schemes for blockchain-based EHRs.

In Table 1, the signing cost of our scheme is refer to the **User-Sign** algorithm. In addition, our scheme also defines that authorities sign the block data, i.e., **Authority-Sign** algorithm, which needs $T_{\mathsf{mtp}} + T_{\mathsf{exp}} \approx 7.079$ ms each time. Furthermore, schemes of [6], [26] have the property of singer privacy based on attribute-based signatures. However, this property is not mandatory for blockchain-based EHRs because, at most time, we need to know who will be responsible for the medical-related data. For example, apparently, patients need to know the identity who gives him the diagnosis.

As described in Subsection IV-D, our scheme supports batch verification which can reduce the verification cost. We divide the batch verification into two cases, verify n signatures from a same signer (denoted by $(1, n)$-to-1 verification) and verify n signatures from n signers (denoted by (n, n)-to-1 verification). Table 2 shows the comparison of efficiency between the three types of verification.

TABLE 1. Comparison of three authentication schemes for blockchain-based EHRs.

	[26]	[6]	Ours
Signing cost	$2tT_{\mathsf{mul}}+$ $(4t+1)T_{\mathsf{par}}+$ $T_{\mathsf{mtp}}+$ $(4t+1)T_{\mathsf{exp}}$ ≈ 244.825 ms	$(6+t)T_{\mathsf{mul}}+$ $NT_{\mathsf{par}}+$ T_{mtp} ≈ 30.614 ms	$T_{\mathsf{mtp}}+$ $2T_{\mathsf{exp}}$ ≈ 12.885 ms
Verification cost	$2tT_{\mathsf{par}}+$ $2tT_{\mathsf{mul}}+$ $(3t+1)T_{\mathsf{exp}}$ ≈ 150.846 ms	$(2tN+1)T_{\mathsf{par}}+$ $T_{\mathsf{mul}}+$ T_{exp} ≈ 301.413 ms	$3T_{\mathsf{par}}+$ $T_{\mathsf{mtp}}+$ $T_{\mathsf{mul}}+$ T_{exp} ≈ 24.498 ms
Communication cost	$\lvert\mathbb{G}\rvert+$ $2\lvert\mathbb{G}_T\rvert$ $= 280$ bytes	$(5+t)\lvert\mathbb{G}\rvert+$ $\lvert\mathbb{G}_T\rvert$ $= 528$ bytes	$2\lvert\mathbb{G}\rvert$ $= 80$ bytes
Resisting collusion attack	No	Yes	Yes

TABLE 2. Efficiency comparison between one-by-one verification (denoted by 1-by-1 verification) and batch verification.

Type of verification	Cost
1-by-1 verification	$3nT_{\mathsf{par}} + nT_{\mathsf{mtp}} + nT_{\mathsf{mul}} + nT_{\mathsf{exp}}$
$(1, n)$-to-1 verification	$3T_{\mathsf{par}} + nT_{\mathsf{mtp}} + T_{\mathsf{mul}} + T_{\mathsf{exp}}$
(n, n)-to-1 verification	$3T_{\mathsf{par}} + nT_{\mathsf{mtp}} + T_{\mathsf{mul}} + nT_{\mathsf{exp}}$

Combining with the run times of basic operations obtained above, we show the time cost of verification algorithm in Figure 4. According to the figure, we can easily observe that the batch verification can significantly reduce the verification delay, especially verifying a large number of signatures.

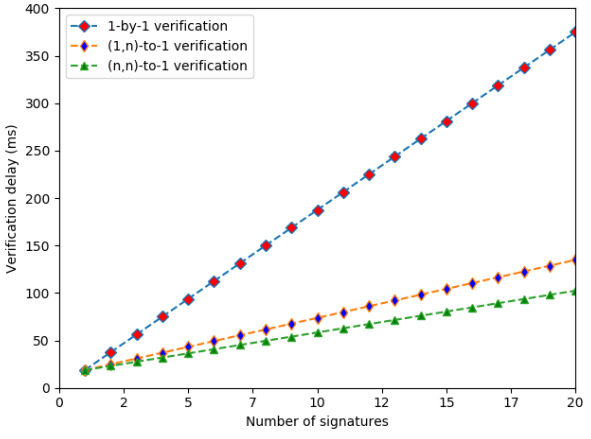

Figure 4. Time comparison of three types of verification.

VI. CONCLUSION

In order to realize the authentication scheme of EHRs system based on blockchain. We first formally define the EHRs system model in the setting of consortium blockchain. Then

we designe an identity-based signature scheme with multiple authorities for the blockchain-based EHRs system. The scheme has efficient signing and verification algorithms.

ACKNOWLEDGMENT

REFERENCES

[1] M. Bellare, J. A. Garay, T. Rabin, "Fast batch verification for modular exponentiation and digital signatures," EUROCRYPT 1998, LNCS 1403, 1998, pp. 236–250.

[2] J. C. Cha, J. H. Cheon, "An identity-based signature from gap Diffie-Hellman groups," PKC 2003, LNCS 2567, 2003, pp. 18–30.

[3] G. G. Dagher, J. Mohler, M. Milojkovic, P. B. Marella, "Ancile: privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology," Sustainable Cities & Society, 2018, 39.

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

[5] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," EUROCRYPT 1999, LNCS 1592, 1999, pp. 295–310.

[6] R. Guo, H. Shi, Q. Zhao, D Zheng, "Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems," IEEE Access, PP 99, 2018, pp. 11676–11686.

[7] H. Li, Y. Dai, X. Lin, "Efficient e-health data release with consistency guarantee under differential privacy," 2015 17th International Conference on E-health Networking, Application & Services (HealthCom), IEEE, 2016: 602–608

[8] H. V. D. Linden, D. Kalra, A. Hasman A, J. Talmon, "Inter-organizational future proof EHR systems: a review of the security and privacy related issues," International Journal of Medical Informatics, Vol. 78(3), 2009, pp: 141–160.

[9] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," IEEE Transactions on Industrial Informatics, vol. 13, no. 6, 2017, pp: 3154–3164.

[10] I. C. Lin, T. C. Liao, "A survey of blockchain security issues and challenges," International Journal of Network Security, vol. 19, no. 5, 2017, pp: 653–659.

[11] H. Lin, J. Shao, C. Zhang, Y. Fang, "CAM: cloud-assisted privacy preserving mobile health monitoring," IEEE Transations on Information Forensics Security, vol. 8, no. 6, 2013, pp. 985–997.

[12] M. Li, S. Yu, Y. Zheng, K. Ren, W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 1, 2013, pp: 131–143.

[13] H. Li, Y. Yang, Y. Dai, S. Yu, Y. Xiang, "Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data, " IEEE Transactions on Cloud Computing, 2017, pp:1-11.

[14] A. Mehmood, I. Natgunanathan, Y. Xiang, H. Poston, Y. Zhang, "Anonymous authentication scheme for smart cloud based healthcare applications," IEEE Access, 2018, pp: 33552-33567.

[15] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," Accessed: 2008. [Online]. Availalbe: https://bitcoin.org/bitcoin.pdf.

[16] U. Premarathne, A. Abuadbba, A. Alabdulatif, I. Khalil, Z. Tari, A. Zomaya, R. Buyya, "Hybrid cryptographic access control for cloud-based EHR systems," IEEE Cloud Computing, vol. 3, no. 4, 2016, pp. 58–64.

[17] D. Pointcheval, J. Stern, "Security arguments for digital signatures and blind signatures," Journal of Cryptology, vol. 13, 2000, pp. 361–396.

[18] A. Shamir, "Identity-based cryptosystems and signature schemes," CRYPTO 1984, Springe, 1985, pp. 47–53.

[19] Standards for Privacy of Individually Identifiable Health Information: Final Rule, Standard 45 CFR Parts 160 and 164, Dec. 2000.

[20] Standard Specification for Continuity of Care Record (CCR), Standard ASTM E2369, 2005.

[21] H. Wang, Y. Song, "Secure cloud-based EHR system using attribute-based cryptosystem and blockchain," Journal of Medical Systems, 2018, vol, 42, no. 8, 152.

[22] W. Xu, L. Wu, Y. Yan, "Privacy-preserving scheme of electronic health records based on blockchain and homomorphic encryption," Journal of Computer Research and Development, vol. 55, no. 10, 2018, pp: 2233–2243.

[23] P. Zhang, J. White, D. C. Schmidt, G. Lenz, S. T. Rosenboloom, "FHIR-Chain: applying blockchain to securely and scalably share clinical data," Computational and Structural Biotechnology Journal, vol. 16, 2018, pp: 267–278.

[24] A. Roehrs, C. A. da Costa, R. da Rosa Righi, "OmniPHR: A distributed architecture model to integrate personal health records," Journal of biomedical informatics, 2017, pp: 70-81.

[25] K. Seol, Y. Kim, E. Lee, Y. Seo, D. Baik, "Privacy-presrving attribute-based access control model for xml-based electronic health record system," IEEE Access, 2018, vol. 6, pp: 9114-9128.

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

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

APPENDIX

PROOF OF LEMMA 1

The proof of lemma 1 is based on the security of the distributed key generation technique [5] which requires that no information on s can be learned by the adversary except for that is implied by the group element $y = g^s \in \mathbb{G}$. The formal definition is as follows which is a variant of the original definition.

Definition 2: For any PPT adversary $\mathcal{A}$, there exists a PPT simulator $\mathcal{S}$, such that on input a random element $y \in \mathbb{G}$ generated by g, produces an output distribution which is polynomially indistinguishable from $\mathcal{A}$'s view of a run of the distributed key generation that ends with y as its public key output, and even if $\mathcal{A}$ corrupts up to $N-1$ authorities.

In $\mathcal{G}_0$, master public key is $y := g^s$ produced by the securely distributed key generation. According to the above definition, even if a PPT adversary $\mathcal{A}$ can corrupt $N-1$ authorities which correspond to the participants in the distributed key generation, it also cannot distinguish between a real key and a random value in the group $\mathbb{G}$. Specifically, any PPT adversary cannot distinguish between $y := g^s \in \mathbb{G}$ and $y := g^{as} \in \mathbb{G}$, where s is the real key produced by the distributed key generation and a, which is independent of s, is the exponent of g^a from a CDH instance even we cannot recover the two exponents s and a. In other words, for any PPT adversary $\mathcal{A}$, its advantages in $\mathcal{G}_0$ and $\mathcal{G}_1$ to break the MA-IBS scheme are identical, i.e., $\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_0}(\lambda)$=$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_1}(\lambda)$.

PROOF OF LEMMA 2

The proof of this lemma is fairly straightforward. In both games $\mathcal{G}_0$ and $\mathcal{G}_1$, the master public key is $y := g^{as}$, where a is the exponent of the element g^a from the CDH instance and s is produced by the distributed key generation. Note that the two values a and s are independently and randomly generated. Therefore, this thing that whether the challenger knows s or not does not affect adversary's view. That is to say, in two games, challenger knows s in $\mathcal{G}_2$ and does not know s in $\mathcal{G}_1$, adversary's advantages to break our MA-IBS scheme are identical, i.e., $\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_1}(\lambda)$=$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_2}(\lambda)$.

PROOF OF LEMMA 3

In $\mathcal{G}_2$, challenger plays the role of all authorities and thus it knows the value s. However, the challenger, in this game, sets the master public key as $y := g^{as}$ rather than g^s that is in the real world. Nevertheless, as proved in Lemma 2, this cannot affect adversary's advantage. We prove that if there exists a PPT adversary $\mathcal{A}$ can break the security of our scheme with a non-negligible advantage $\epsilon(\lambda)$, then we can construct an efficient algorithm, i.e., challenger $\mathcal{C}$, to solve the CDH problem with a non-negligible probability $\epsilon'(\lambda)$.

Firstly, challenger $\mathcal{C}$ is given an instance $(\mathbb{G}, \mathbb{G}_T, p, g, e, g^a, g^b)$ of CDH problem. The goal of $\mathcal{C}$ is to compute $g^{ab} \in \mathbb{G}$. Then, $\mathcal{C}$ plays the following game with the adversary $\mathcal{A}$.

- **System Setup**: Given a security parameter λ, $\mathcal{C}$ executes the system setup algorithm on the behalf of the EHR server as in the real world. The output of this phase is the public parameters $params$. Then, $\mathcal{C}$ gives it to $\mathcal{A}$.
- **Authority Setup**: $\mathcal{C}$ honestly runs the authority setup algorithm on the behalf of all authorities $ID_1, ..., ID_N$. The output of this phase is the master secret key $SK_{ID_1}, ..., SK_{ID_N}$. $\mathcal{A}$ can obtain identities $ID_1, ..., ID_N$. Note that, in such setting, $\mathcal{C}$ plays the roles of all authorities, and hence it knows the secret value $s \in \mathbb{Z}_p^*$. Then, $\mathcal{C}$ sets $y := g^{as}$ which means that the implied master secret key is as. Note that this is different from that in the real scheme, but according to Lemma 2, it is indistinguishable to $\mathcal{A}$. Finally, $\mathcal{C}$ adds parameters y and $\{A_{i0}\}_{i\in[1,N]}$ into $params$ and gives $params := \{\mathbb{G}, \mathbb{G}_T, p, g, y, e, H, H_1, \{(ID_i, A_{i0})\}_{i=1}^N\}$ to $\mathcal{A}$, where H and H_1 will be seem as the random oracles in the following proof.
- **Queries**: $\mathcal{A}$ makes the following queries to $\mathcal{C}$:
  - H random oracle: challenger $\mathcal{C}$ maintains a list $L := \{(i, id_i, h_i' \in \mathbb{G})\}$ where
  $$h_i' = \begin{cases} g^b, & \text{if } i = i^* \\ g^{k_i}, & \text{if } i \neq i^* \end{cases}$$
  and i^* is randomly selected by $\mathcal{C}$ which denotes its guessing point that $\mathcal{A}$ will attack and k_i is also randomly selected by $\mathcal{C}$ from $\mathbb{Z}_p^*$. If $\mathcal{A}$'s query id_i in the list L, then $\mathcal{C}$ gives h_i' to $\mathcal{A}$; Else, $\mathcal{C}$ randomly selects an integer $k_i \in \mathbb{Z}_p^*$, then gives $h_i' = g^{k_i} \in \mathbb{G}$ to $\mathcal{A}$ and adds the item (i, id_i, h_i') into the list L.
  - H_1 random oracle: challenger $\mathcal{C}$ also maintains another list $L_1 := \{(i, m_i||u_i, h_{1i}')\}$ where $h_{1i}' \in \mathbb{Z}_p^*$ is randomly chosen by $\mathcal{C}$. If $\mathcal{A}$'s query $m_i||u_i$ in the list L_1, then $\mathcal{C}$ gives h_{1i}' to $\mathcal{A}$; Else, $\mathcal{C}$ randomly selects an integer $h_{1i}' \in \mathbb{Z}_p^*$, then gives h_{1i}' to $\mathcal{A}$ and adds the item $(i, m_i||u_i, h_{1i}')$ into the list L_1.
  - Master secret key generation oracle: $\mathcal{A}$ issues a request for some corrupted authorities $ID_{i\in Q_M} \subset \{ID_1, ..., ID_N\}$ for their master secret key, where Q_M denotes the index set of the identities of corrupted authorities. For such a request, $\mathcal{C}$ transmits $SK_{ID_{i\in Q_M}}$ to $\mathcal{A}$. As described above, the master secret keys were honestly generated by $\mathcal{C}$ alone, therefore it can answer such queries.
  - Key generation oracle: $\mathcal{A}$ submits an identity id_i to $\mathcal{C}$ for its secret key sk_{id_i}. If $id_i = id_{i^*}$, then $\mathcal{C}$ cannot answer this query and thus has to abort the game. Else, $\mathcal{C}$ returns back $sk_{id_i} = (g^a)^{s\cdot k_i} \in \mathbb{G}$, where k_i is from the list L, to $\mathcal{A}$.
  - User-Signing oracle: $\mathcal{A}$ submits a tuple (id_i, m_i) to $\mathcal{C}$ for a signature σ_i. The request of $\mathcal{A}$ can be divided into two cases:
    1) $id_i \neq id_{i^*}$: in such case, $\mathcal{C}$ firstly obtains a valid secret key sk_{id_i} from the key generation oracle. Then, it can compute the signature for the query (id_i, m_i) as the real signer did in the real world.
    2) $id_i = id_{i^*}$: in such case, $\mathcal{C}$ cannot obtain a valid secret key $sk_{id_{i^*}}$ from the key generation oracle and thus it cannot directly compute the signature as above. However, $\mathcal{C}$ also can answer $\mathcal{A}$'s request as follow: $\sigma_i := (u_i, v_i) = (g^{e_i}/(g^b)^{h_{1i}'}, (g^a)^{e_i\cdot s})$ where h_{1i}' is from the H_1 random oracle and e_i is randomly selected from $\mathbb{Z}_p^*$. The correctness of the signature can be verified as follows:
    $$\begin{aligned} & e(u_i, y) \cdot e(H(id_i)^{t_j}, y) \\ = \; & e(g^{e_i}/(g^b)^{h_{1i}'}, g^{as}) \cdot e((g^b)^{h_{1i}'}, g^{as}) \\ = \; & e(g^{e_i}, g^{as}) \\ = \; & e(g^{e_i\cdot as}, g) \\ = \; & e(v_i, g). \end{aligned}$$
- **Forgery**: $\mathcal{A}$ outputs a forgery $(m^*, \sigma^* = (u^*, v^*))$ with identity id^*. We assume that $\mathcal{A}$ wins the game, that is to say, the forgery satisfies all of the following conditions:
  1) $Q_M \neq [1, N]$;
  2) **Verify**$(id^*, m^*, \sigma^*) = Accept$;
  3) $\mathcal{A}$ does not make query for identity id^* in the key generation queries phase;
  4) $\mathcal{A}$ does not make query for message m^* and identity id^* in the signing queries phase.

  In addition, we assume that the advantage of $\mathcal{A}$ to win the above game is ϵ.

After the end of the game, $\mathcal{C}$ checks that $id^* \stackrel{?}{=} id_{i^*}$. If the equation does not holds, then $\mathcal{C}$ aborts the game and outputs $\perp$. Otherwise, according to the forking lemma [17], based on the valid signature $(id^*, m^*, u^*, h_1^* \leftarrow H_1(m^*||u^*), v^*)$, $\mathcal{C}$ also can obtain another valid signature $(id^*, m^*, u^*, h_1^{*'} \leftarrow H_1(m^*||u^*), v^{*'})$ with probability $\frac{1-e^{-1}}{q_{H_1}}$, where $h_1^* \neq h_1^{*'}$, by rewind the random oracle with the same input $m^*||u^*$ but different choices of H_1.

Finally, according to the assumption that $\mathcal{C}$ obtains two valid signatures from $\mathcal{A}$, it can compute

$$g^{ab} = \left(\frac{v^*}{v^{*'}}\right)^{s^{-1}\cdot(h_{1i}^* - h_{1i}^{*'})^{-1}} \pmod p.$$

which is the desired solution of the given instance of CDH problem.

According to the above proof, a successful simulation, denoted by E, from $\mathcal{C}$ consists of three events:

- E_1: in key generation oracle, $\mathcal{A}$ does not query id_{i^*}'s secret key.
- E_2: adversary $\mathcal{A}$'s challenge identity is id_{i^*}.
- E_3: challenger $\mathcal{C}$ successfully rewinds the random oracle H_1 for the forking lemma.

A successful simulation means that $\mathcal{C}$ can use $\mathcal{A}$'s forgery to solve the CDH problem. In other words, the advantage of $\mathcal{C}$ is $\mathbf{Adv}_A^{CDH}$=Pr[E]=Pr[E_1]·Pr[E_2]·Pr[E_3] since the three events are mutually independent. Therefore, the advantage of $\mathcal{C}$ to solve the CDH problem is

$$\mathbf{Adv}_C^{CDH} = \epsilon' \geqslant \prod_{i=1}^{q_K} \frac{q_H - i}{q_H} \cdot \frac{1}{q_H} \cdot \frac{1 - e^{-1}}{q_{H_1}} \cdot \epsilon.$$

PROOF OF THEOREM 1

According to the three lemmas, we can see that the advantages in three games $\mathcal{G}_0$, $\mathcal{G}_1$ and $\mathcal{G}_2$ of a PPT adversary to break the scheme are identical, i.e., $\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_0}$=$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_1}$=$\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_2}$. In addition, its advantage in $\mathcal{G}_2$ is negligible. Therefore, its advantage in $\mathcal{G}_0$, $\mathbf{Adv}_{\mathcal{A}}^{\mathcal{G}_0}(\lambda)$, which corresponds to the real world of when using our MA-IBS scheme into the blockchain-based EHRs system, is also negligible.

Let T_{exp} be the time to perform exponentiation operations in group $\mathbb{G}$. Assume that a $(t, q_H, q_{H_1}, q_M, q_K, q_S, \epsilon)$-adversary successfully breaks this MA-IBS scheme. According to the proof of Lemma 3, then there exists efficient algorithm $\mathcal{C}$ to solve the CDH problem with time $t' \approx t + q_H T_{\mathsf{exp}} + q_K T_{\mathsf{exp}} + 2q_S T_{\mathsf{exp}} + T_{\mathsf{dkg}}$, where T_{dkg} denotes the time of $\mathcal{C}$ to simulate the distributed key generation algorithm.

PROOF OF THEOREM 2

Challenger $\mathcal{C}$ is given an instance, $(\mathbb{G}, \mathbb{G}_T, p, g, e, g^a, g^b)$, of CDH problem and its target is to compute $g^{ab} \in \mathbb{G}$. $\mathcal{C}$ plays the following game with adversary $\mathcal{A}$.

- **System Setup**: $\mathcal{C}$ randomly chooses a security parameter λ and executes the system setup algorithm. Then it outputs the public system parameters to $\mathcal{A}$.
- **Authority Setup**: Next, $\mathcal{C}$ honestly executes the authority setup algorithm on the behalf of all authorities $ID_1, ..., ID_N$ with the exception of $SK_{ID_{i^*}} := a$ even $\mathcal{C}$ does not know it, where i^* is randomly chosen by $\mathcal{C}$ which denotes its guess of $\mathcal{A}$'s challenge authority. Finally, $\mathcal{C}$ adds parameters y and $\{A_{i0}\}_{i\in[1,N]}$, where $A_{i^*0} := g^a$, into $params$ and gives it to $\mathcal{A}$.
- **Queries**: $\mathcal{A}$ makes the following queries to $\mathcal{C}$:
  - H random oracle: challenger $\mathcal{C}$ maintains a list $L := \{(i, M_i, h_i' \in \mathbb{G})\}$ where

$$h_i' = \begin{cases} g^b, & \text{if } i = i^{*'} \\ g^{k_i}, & \text{if } i \neq i^{*'} \end{cases}$$

    and $i^{*'}$ is randomly selected by $\mathcal{C}$ which denotes its guessing point that $\mathcal{A}$ will attack and k_i is also randomly selected by $\mathcal{C}$ from $\mathbb{Z}_p^*$. If $\mathcal{A}$'s query M_i in the list L, then $\mathcal{C}$ gives h_i' to $\mathcal{A}$; Else, $\mathcal{C}$ randomly selects an integer $k_i \in \mathbb{Z}_p^*$, then gives $h_i' = g^{k_i} \in \mathbb{G}$ to $\mathcal{A}$ and adds (i, M_i, h_i') into the list L.
  - H_1 random oracle: challenger $\mathcal{C}$ also maintains another list $L_1 := \{(i, m_i||u_i, h_{1i}')\}$. If $\mathcal{A}$'s query in the list L_1, then $\mathcal{C}$ gives h_{1i}' to $\mathcal{A}$; Else, $\mathcal{C}$ randomly selects an integer $h_{1i}' \in \mathbb{Z}_p^*$, then gives it to $\mathcal{A}$ and adds the item $(i, m_i||u_i, h_{1i}')$ into the list L_1.
  - Master secret key generation oracle:$\mathcal{A}$ submits an authorities ID_i to $\mathcal{C}$ for its master secret key SK_{ID_i}. If $ID_i = ID_{i^*}$, then $\mathcal{C}$ cannot answer this query and thus has to abort the game. Else, $\mathcal{C}$ returns back the real SK_{ID_i} to $\mathcal{A}$.
  - Key generation oracle: $\mathcal{C}$ knows the value of a_{j0} for $j = 1, \cdots, N$ and $j \neq i^*$ and thus can compute $psk_{id_i,j} = H(0||id_i)^{a_{j0}}$. For psk_{id_i,i^*}, $\mathcal{C}$ does not know a, but it can retrieval the exponent of $H(0||id_i) = g^{k_i}$ from the H random oracles and thus it also can compute $psk_{id_i,i^*} = (g^a)^{k_i}$. Finally, $\mathcal{C}$ sends $sk_{id_i} = H(0||id_i)^s = \prod_{i=1}^{N} psk_{id_i,j}$, for $j = 1, \cdots, N$.
  - Authority-Signing oracle: $\mathcal{A}$ submits a message M_i to $\mathcal{C}$ for a signature σ_i. The request of $\mathcal{A}$ can be divided into two cases:
    1) $ID_i \neq ID_{i^*}$: in such case, $\mathcal{C}$ firstly obtains a valid secret key SK_{ID_i} from the master secret key generation oracle. Then, it can compute the signature for the query (ID_i, M_i) as the real signer did in the real world.
    2) $ID_i = ID_{i^*}$: in such case, if $\mathcal{A}$'s query $M_i \neq M_{i^{*\prime}}$, it can compute the signature for the query (ID_{i^*}, M_i) as $\delta_i = (g^a)^{k_i}$. If $M_i = M_{i^{*\prime}}$, then $\mathcal{C}$ cannot answer this query and thus has to abort the game.
- **Forgery**: $\mathcal{A}$ outputs a forgery (M^*, σ^*) with identity ID^*. We assume that $\mathcal{A}$ wins the game, that is to say, the forgery satisfies all of the following conditions:
  1) **Verify**$(ID^*, M^*, \sigma^*) = Accept$;
  2) $i^* \notin Q_M$;
  3) $\mathcal{A}$ does not make query for message $M_{i^{*\prime}}$ and identity ID_{i^*} in the signing queries phase.

  In addition, we assume that the advantage of $\mathcal{A}$ to win the above game is $\mathbf{Adv}_{MA-IBS,\mathcal{A}}^{UF-CMA}(\lambda) = \epsilon$. According to the above proof, a successful simulation, denoted by E, from $\mathcal{C}$ consists of three events:
  - E_1: in master secret key generation oracle, $\mathcal{A}$ does not query ID_{i^*}'s master secret key.
  - E_2: in signing oracle, $\mathcal{A}$ does not query $(ID_{i^*}, M_{i^{*\prime}})$'s signature.
  - E_3: adversary $\mathcal{A}$'s challenge identity is ID_{i^*} and message is $M_{i^{*\prime}}$.

A successful simulation, apparently, means that $\mathcal{C}$ can use $\mathcal{A}$'s forgery to solve the CDH problem. In other words, the advantage of $\mathcal{C}$ to solve the CDH problem is $\mathbf{Adv}_{\mathcal{A}}^{CDH}$=Pr[$E$]=Pr[$E_1$]·Pr[$E_2$]·Pr[$E_3$] since the three events

are mutually independent. Therefore, the advantage of $\mathcal{C}$ to solve the CDH problem is

$$\mathbf{Adv}_{\mathcal{C}}^{CDH} = \epsilon' \geqslant \prod_{i=1}^{q_M} \frac{N-i}{N} \cdot (\prod_{i=1}^{q_{AS}} \frac{N-i}{N} + \prod_{i=1}^{q_{AS}} \frac{q_H-i}{q_H} - \prod_{i=1}^{q_{AS}} \frac{(N-i)(q_H-i)}{N \cdot q_H}) \cdot \frac{1}{N} \cdot \frac{1}{q_H} \cdot \epsilon.$$

According to the above, there exists efficient algorithm $\mathcal{C}$ to solve the CDH problem with time $t' \approx t + q_H T_{\mathsf{exp}} + q_K T_{\mathsf{exp}} + q_S T_{\mathsf{exp}} + T_{\mathsf{dkg}}$.