IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 17, 2019, accepted October 10, 2019, date of publication October 14, 2019, date of current version October 25, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2947313*

# MBPA: A Medibchain-Based Privacy-Preserving Mutual Authentication in TMIS for Mobile Medical Cloud Architecture

**XIAOXUE LIU[1], WENPING MA[1], AND HAO CAO[2]**
[1]State Key Lab of Integrated Service Network, Xidian University, Xi'an 710071, China
[2]School of Information and Network Engineering, Anhui Science and Technology University, Chuzhou 233100, China

Corresponding author: Xiaoxue Liu (862417756@qq.com)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802400, in part by the Fundamental Research Funds for the Central Universities and the Innovation Fund of Xidian University under Grant 5001-20109195456, in part by the National Science Foundation of China under Grant 61373171, in part by The 111 Project under Grant B08038, and in part by the Program for Excellent Young Talents in University of Anhui Province under Grant gxyqZD2019060.

**ABSTRACT** Telecare Medical Information System (TMIS) integrates various types of mobile devices and communication technologies to upgrade the traditional face-to-face medical treatment model to intelligent one, which can provide the flexible and convenient e-health care. Due to the complexity and openness of Internet, e-health care data is grabbing the interest of cyber attackers. Hence, security and privacy are still our dominant concerns. Fortunately, blockchain technology leverages decentralized or distributed process to ensure data security. A MediBchain-based privacy-preserving mutual authentication for mobile medical cloud architecture (abbreviated to MBPA) is proposed in this paper. MBPA scheme not only mitigates the weaknesses of existing ones, but has other advantages. First, MBPA scheme supports patients anonymity and traceability since the patient's identity is hidden in two dynamic anonyms and a static anonym and only the trusted center can recover his/her real identity. Second, each MediBchain node shares a secret value, which realizes authentication with extremely low computional cost between terminals and MediBchain nodes. Finally, MBPA scheme is proven safely against passive and active attacks under elliptic curve computational Diffie-Hellman problem (ECDHP) assumption in random oracle model. Hence, these features make MBPA scheme very suitable for computation-limited mobile devices compared with other related existing schemes.

**INDEX TERMS** TMIS, MediBchain, cloud, privacy-preserving, traceability.

## I. INTRODUCTION

Telecare Medical Information System (TMIS) for mobile medical cloud architecture is new paradigm of blending medical treatment, blockchain, cloud computing and cloud storage together via the Internet to provide high quality e-health care for terminals (patients/doctors). Recently, TMIS has attracted a lot of works in an amalgamated manner, which brings enormous changes to e-health care paradigm. These changes not only affect treatment process of the patient, but also affect the diagnostic reasoning of the doctor. In TMIS, patients are monitored by electronic medical devices or tiny sensors to collect their vital signs and other health data [1]. Then, they send these signs and data to cloud for later access by doctors. Based on these signs and data, doctors give the corresponding diagnosises and send the corresponding treatment plans to cloud for later access by patients. Hence, patients can receive professional diagnosises, directly and timely.

As is well known, e-health care is completely dependent on these signs and data. Hence, it is essential to ensure these data integrity and authenticity. However, patients' data and doctors' diagnosis data are transmitted and exposed during the unsecured public communication channel. Adversaries may eavesdrop, intercept, delete, and modify all the data in this channel, easily. Cloud may eavesdrop, intercept, delete, and modify all the data in his own database, which is often difficult to detect. Considering the worst condition, if the adversary has an attempt at harming the patient, he may modify this patient's vital health data. Once, these modified data is transmitted to his doctors, the wrong diagnosis can

The associate editor coordinating the review of this manuscript and approving it for publication was Minho Jo.

be made and the patient's life may be threatened [2]. The data stored in cloud is not complete or it doesn't exist over time. Not surprisingly, they are the often case in our daily routine. Obviously, the patients' privacy protection has not been adequately addressed and securely sharing e-health care information is still an urgent and much stronger requirement in TMIS.

Motivated by the recent explosion of interest around blockchain in the Internet of Things (IoT) sector, it also has attracted the interest of stakeholders across a wide span of industries, especially in the TMIS [3]. The blockchain technology allows participants to move data in real-time, without exposing the channels to theft, forgery and malice. Each transaction stored in the blockchain has a corresponding hash. Meanwhile, this hash value generates a binary Merkle tree, which is stored in the block header together with a timestamp and the identifier of the previous block. Therefore, if an attacker wants to tamper with a record in the blockchain, he needs not only to modify the hash of the block, but also to modify the hash of all subsequent blocks which are nearly impossible to achieve. What's more, the transaction data stored in the blockchain contains not only the hash value, but also the signature of both parties which is unforgeable. Further, based on these hash values in Merkle tree, patients and docrors can discover whether the data stored in the cloud have been tampered with. In the blockchain system, the user performs a series of hash operations on the public key and obtains a fixed-length hash value as the corresponding account in order to cut off the connection between the real identity [4]. Hence, blockchain technology can greatly improve patient privacy leakage during the process of seeking e-health care and offer patients and doctors the abilities to securely share e-health care information across these platforms. In the past, we imagined a future where patients could hold the keys to their e-health care passport and imagined a better quality of e-health care for both patients and doctors [5]. Relying on blockchain technology, all thhese will come ture.

To TMIS, blockchain technology has several significant advantages:(a)Its decentralized distributed structure helps us to share health data; (b)Untamperable timestamp can help us to solve the tracing problem of data and equipment and the problem of the information anti-counterfeiting; (c)Its advantages of high redundancy and complex custody rights of multiple private keys can solve the security authentication defects of current medical information technology;(d)Its flexible programmable features can help hospitals build and expand their applications [10]–[13]. These features make blockchain more suitable for medical scenarios. Specifically, blockchain enable us to have a distributed peer-to-peer network, where untrustworthy members still can interact with each other without a trusted center in a verifiable way.

However, the existing related schemes require massive interactions between the terminals' gateway devices and each blockchain node to negotiate a secret key, which consumes a large number of transmission and computation costs. Considering the low battery supply and weak computing capability of the tiny medical sensors, they are not suitable for computation-limited mobile devices. Hence, it is urgent to design a scheme, which is very suitable for computation-limited mobile devices.

In MBPA scheme, we resolve above discussed problems by storing the encrypted data in cloud based on medical blockchain (abbreviated to MediBchain) technology. Additionally, it is desirable to generate a secret shared key (among MediBchain nodes) without key negotiation rounds to ensure the security, such as the source authentication. After the shared key is distributed, the terminals encrypt their signs and data using the shared public key and broadcast them to MediBchain nodes network. To resist impersonation attack, the patient authenticates these cloud ciphertexts sent by doctors in his/her own TMIS. Batch verification method should be designed to accelerate the authenticating speed.

### A. MAIN CONTRIBUTIONS

The main contribution of this research is the conceptual MediBchain-based system for mobile medical cloud architecture.

- In TMIS, the identities of patients are sensitive and may leak patients' privacy. Compared with traditional blockchains technology, in MediBchain technology, both MediBchain and dynamic encrypted messages are integrated to anonymously authenticate terminals. The patient performs a series of hash operations on two dynamic anonyms $PID^1_{PA_i}$ and $PID^2_{PA_i}$ and a static anonym $pid_{PA_i}$ and obtains a corresponding fixed-length hash value as the corresponding account in order to cut off the connection between the real identity, which is different in each run. If a patient $PA_i$ is compromised and utilized to launch attack in MBPA scheme, the trusted center ($TC$) can also recover his/her real identity from the static anonym $pid_{PA_i}$.
- For the multi-receivers mode in MediBchain technology, secret sharing is utilized to provide confidentiality. In other words, only authorized participates (i.e. registered at $TC$) can get the secret shared value to obtain the legal ciphertext of request messages. What's more, the secret value shared among MediBchain nodes without key negotiation rounds undoubtedly greatly accelerates the authenticating speed in comparison with traditional blockchains model. In MBPA scheme, after the secret shared key distributed, the terminals encrypt their signs and data using the shared public key and broadcast them to MediBchain nodes network without multiple encryptions. A large amount of ciphertexts may arrive at the same time period, MBPA scheme still provides a batch verification algorithm to improve the efficiency. Hence, MBPA scheme is very suitable for big data medical system.
- In order to guarantee the integrity, timeliness and confidentiality of the data stored in the cloud, we design a novel feedback (response) mechanism based on terminals' temporary keys, which are only obtained by terminals and cloud. When the terminals successfully store a transaction into the cloud, cloud feedbacks the results to terminals in a timely manner.

• MBPA scheme based on certificateless cryptography can overcome the key escrow problem of identity-based public key cryptography. The full private keys of terminals (patients/doctors) consist of two parts: the secret values chosen by terminals themselves and the partial secret keys generated by trusted center (*TC*). It properly resolves the complicated certificate management problems in traditional public key infrastructure system.

• MBPA scheme is proved to be secure under the Elliptic Curve Computational Diffie-Hellman problem (ECDHP) (For details, see 3.1) assumption in the random oracle model. The MBPA scheme is proved secure against possible known attacks and satisfy the secure requirements of AKA scheme. Hence, the MBPA scheme is practical in complex network environment.

### B. RELATED WORK

Data collection and data delivery are the key points in TMIS. In order to better collect, deliver and store data, some schemes based on cloud for TMIS have been proposed in [2], [6]–[13]. Rolim *et al.* [6] proposed a model, where the system processes the data during data collection and data delivery. In this model, the sensor played the role as a collector. Due to limited computational capability, the energy and bandwidth resource constraints of the sensor nodes, the effects of this model are not satisfactory in practice. Later, a cloud-based patient privacy data processing model was proposed in [7]. Initially, the model was designed to solve the problem of difficult medical treatment in rural areas. However, in order to ensure cost-effectiveness, patients in rural areas needed to bear tremendous economic pressure. Therefore, it is not suitable for real-life situation. Hence, their model also has not been widely used. In order to help people better understand the patient-centered cloud-based TMIS, Zhang *et al.* [8] only introduced the system model in detail and did not give specific solutions. Based on the research of pioneers, Chen *et al.* [9] constructed a authentication scheme by using cloud computing and e-health services to provide medical service. They claimed their scheme was secure against many common attacks. However, the scheme was pointed out that it could not guarantee patient's anonymity and message authentication [10]. Later, [11], [12] stated that the scheme in [10] was still failed to provide patient's anonymity and message confidentiality. It also suffered from KCI attack. Reference [13] found that the scheme in [12] could not protect patient anonymity and unlinkability. Unfortunately, Liu and Ma [2] found that the schemes [11]–[13] still suffered from some security threats. In [11], [12], each valid patient could easily obtain the cloud server's private key. It is no doubt to increase secure risk of the system. For [13], it also could not ensure patient's anonymity and message confidentiality. However, in [2], [9]–[13], the cloud not only participates in the authentication but also needs to store data, which causes a certain time delay, which is not conducive to the timely diagnosis of the patient.

In order to continuously improve TMIS and motivated by the recent explosion of interest around blockchains, Linn and Koo [14] proposed a blockchain based access control manager for heath data for enhancing the inter-operability. Yue *et al.* [15] proposed a data access control model based on blockchain. Xu *et al.* [16] showed a case study, where the study showed their experience of building originChain. Simic *et al.* [17] showed a case study, where they introduced the remarkable advantages of the combination of the IoT and blockchains. Ekblaw *et al.* [18] proposed a prototype named 'MedRec' to solve the security issues solution for electronic health records (EHR)by using blockchain. Chen *et al.* [19] designed a storage scheme to manage personal medical data based on blockchain and cloud storage in theory. Dwivedi *et al.* [20] took an initial look at a blockchain-based IoT model glimpsing into an advanced security and privacy model to be used in any current IoT-based remote monitoring system. The above literatures are described from the general framework, and no specific plans are given.

In other systems, there were some specific blockchain-based authentication protocols. Wang *et al.* [21] proposed a blockchain-based mutual authentication security protocol. In this new scheme, there was no need for the trusted third parties to provide security and privacy. Reference [22] designed a novel privacy-preserving incentive announcement network based on blockchain via an efficient anonymous vehicular announcement aggregation protocol. Conti *et al.* [23] proposed a blockchain based lightweight distributed mobile producer authentication (BlockAuth) protocol to enable secure and efficient mobility management in information centric networking (ICN). Reference [24] proposed a blockchain-based fair nonrepudiation service provisioning protocol for IIoT scenarios in which the blockchain was used as a service publisher and an evidence recorder. Kim *et al.* [25] proposed a secure charging protocol for electric vehicles based on blockchain to resolve these security flaws. Reference [26] introduced a blockchain based mutual authentication and key agreement protocol for edge computing based smart grid systems. Wang *et al.* [27] proposed the novel concept of blockchain-based anonymous reporting protocol with anonymous rewarding for the first time.

Based on the previous achievements, Guo *et al.* [28] proposed a flexible and efficient blockchain-based ABE prptocol with multi-authority for medical on demand in telemedicine system. However, in patient's side, the patient should remember a tuple of private keys, which is unsuitable for older people. Tang *et al.* [29] designed an identity-based protocol with multiple authorities for the blockchain-based EHRs system. The protocol has efficient signing and verification algorithms. Reference [30] also stated a blockchain based searchable encryption for electronic health record sharing protocol, where the data owners have full control over who can see their EHRs data. Mohsin *et al.* [31] constructed a novel verification secure protocol for patient authentication based blockchain-PSO-AES techniques in finger vein biometrics, which is not suitable for computation-limited

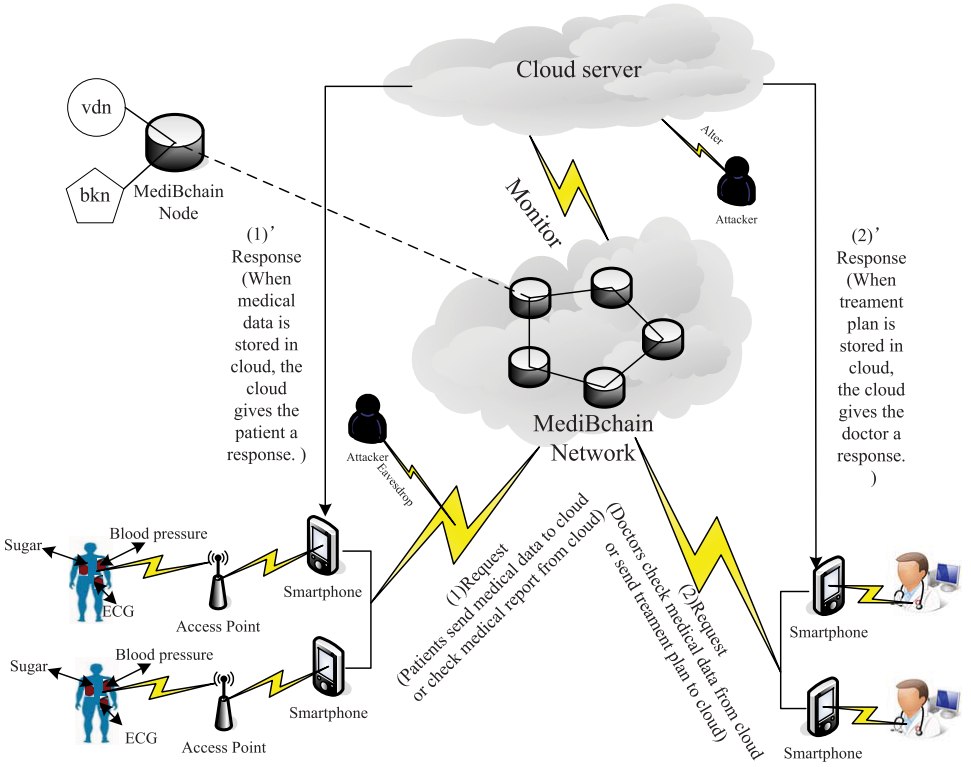



**FIGURE 1.** Architecture for accessing medical care in MediBchain Network.

mobile devices. The above literatures are described for EHRs system, which could not solved technique problems of TMIS in true sense.

Based on their work, Alomar *et al.* [3] proposed a specific scheme: A blockchain based privacy preserving platform for healthcare data, where the authentication process relies on secure channels, which is not in practice.

### C. ORGANIZATION

The rest of paper is organized as follows. Some system models about MBPA scheme are introduced in Section 2. Section 3 briefly reviews system building blocks and the MBPA scheme is presented in Section 4. Detailed security analysis and proof are given in Section 5. The comparisons of the performance and security features between MBPA scheme with other related schemes are discussed in Section 6. Section 7 concludes this paper.

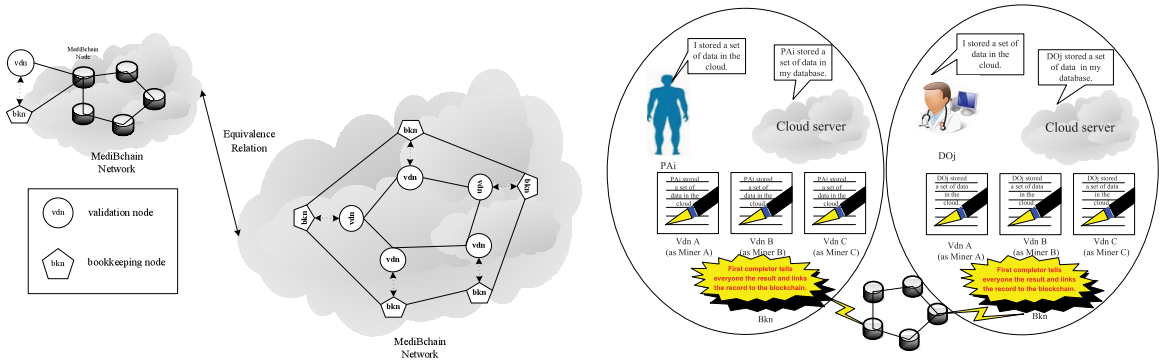
## II. SYSTEM MODELS

In this section, we introduce the system model. The participants involved in MBPA scheme include Terminals (patients/doctors), MediBchain Network and Cloud as shown in Fig.1.

- *Terminals:* Terminals are often some remote mobile devices of patients and doctors (e.g. Smartphone and iPad) that can request or access commands remotely. When these terminals wish to ask or answer for a medical request, they need to brostcast a corresponding transaction messages to the MediBchain network.
- *MediBchain network:* In MBPA scheme, we utilize some trusted permissioned nodes (i.e. registered at *TC*) responsible for maintaining the MediBchain. Furthermore, we utilize a practical consensus mechanism (PBFT) to maintain the MediBchain, which is unlike the Bitcoin blockchain. The Bitcoin blockchain is only used to proof of work. For detailed properties of PBFT, please refer to 3.2. Permissioned nodes are divided into "validation node" (working as a miner) and "bookkeeping node",as shown in Fig.2(a). In MBPA scheme, "validation node" is responsible for verifying transactions (abbreviated as "vdn") and "bookkeeping node" takes charge of chaining validated transactions into the MediBchain (abbreviated as "bkn"), which only accepts valid transactions sent by its corresponding "vdn", as shown in Fig.2(b).
- *Cloud:* The cloud hosts a large number of information(e.g. medical report), which collects and processes massive data from terminals (patients/doctors). Meanwhile, it also responds to the access data requests from terminals. That is, the cloud monitors the MediBchain network and responds to the requested data or terminals.

## III. SYSTEM BUILDING BLOCKS

In this section, we review some cryptography materials used in MBPA scheme and introduce a practical consensus mechanism (PBFT).




(a) Architecture for MediBchain Node

(b) Terminals access MediBchain nodes and MediBchain

**FIGURE 2.** MediBchain network.

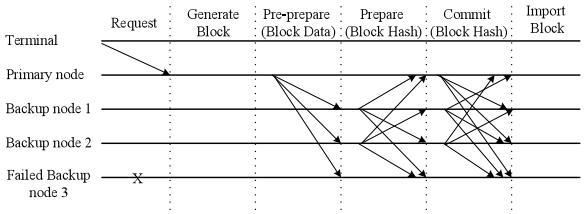



**FIGURE 3.** Practical byzantine fault tolerance (PBFT).

### A. ELLIPTIC CURVE COMPUTATIONAL DIFFIE-HELLMAN PROBLEM (ECDHP)

Choose $G_p$ as elliptic curve group $G_p$ with the generator of $P$, whose order is a prime $p$. Given $(P, aP, bP) \in G_p$ for any unknown $a, b \in Z_p^*$, the goal of the ECDHP is to compute $abP$. Define the advantage of any probabilistic polynomial time algorithm $\mathcal{A}$ against ECDHP in $G_p$. For every probabilistic $\mathcal{A}$, the advantage is negligible, which will be used in the security analysis of MBPA scheme.

### B. PRACTICAL BYZANTINE FAULT TOLERANCE (PBFT)

Practical Byzantine Fault Tolerance (PBFT) is a consensus mechanism algorithm [32], [33]. In this algorithm, after messages are exchanged among distributed network nodes, each node lists all the information they received. Finally, most of the same messages are used as the original messages. Specifically, PBFT is realized mainly by the decision of quorum, where one node represents one vote. The maximum fault tolerance is not more than 1/3 of the total number of nodes, which means that if there are more than 2/3 of the honest nodes, the whole system will be able to operate normally, where $Z > 3F + 1$, $Z$ is the total number of nodes and $F$ is total number of problematic nodes.

PBFT mainly consists of the following phases, which is adopted in MBPA scheme, which mainly consists of the following phases (see Fig.3).

• *Generate Block:* A primary node is in charge of generating a new candidate block. According to the storage capacity and computing capacity, each consensus nodes may be primary node to generate the block in MBPA scheme.

• *Pre-prepare (Block Data):* Upon receiving a request from a terminal, the primary node generates pre-prepare message $\{(Pre-prepare, vn, sq, md), m\}$ (Here, $ve$ is view number; $sq$ is the order number of request message in view $vn$; $md$ is the message digest of request message; $m$ is the request message)and broadcasts the pre-prepare message to other consensus nodes. Pre-prepare message is intended to prove that $sq$ is given by the primary node in $ve$.

• *Prepare (Block Hash):* After receiving pre-prepare messages, each consensus nodes (including primary node) will enter prepare phase. Each one certifies their validity, generates prepare message $\{(Prepare, vn, sq, md), i\}$ ($i$ is node number), stores the prepare message into a message log and broadcast the prepare message with the hash value to the other nodes. Obviously, each consensus nodes stores at least $2F+1$ prepare message.

• *Commit (Block Hash):* After receiving sufficient prepare messages (i.e. the number of messages is more than $2F + 1$), each consensus nodes (including primary node) will enter commit phase. Each one certifies their validity, generates commit message $\{(Commit, vn, sq, md), i\}$ and broadcasts the commit message to the other ones.

• *Import Block:* Any $F + 1$ normal consensus nodes reach a same consensus on the candidate block if the number of received commit messages is not less than $2F + 1$. That is, the block can be chained into the permissioned MediBchain.

## IV. MBPA: A MEDIBCHAIN-BASED PRIVACY-PRESERVING MUTUAL AUTHENTICATION

The MBPA is composed of twelve basic phases: **Global setup phase**, **Patient registration phase**, **Doctors registration phase**, **MediBchain nodes registration phase**, **MediBchain node secret shared value setting phase**, **Patient request phase**, **Chain transaction phase (I)**, **Storage and response phase(I)**, **Treatment phase**, **Chain transaction phase (II)** **Storage and response phase(II)**, and **Checking up phase**. To simplify the subsequent description, some symbol notations are given in **Table 1**. **Fig.1.** simply depicts the MediBchain network model. At the beginning, trusted center $TC$ sets up its systems:

**TABLE 1.** Notations.

| Symbol | Description |
|---|---|
| $ID_{PA_i}, ID_{DO_k}$ | the identity of Patient($PA_i$) and Doctor($DO_k$) |
| $pid_{PA_i}$ | the static anonym of $PA_i$ |
| $PID^1_{PA_i}, PID^2_{PA_i}$ | the dynamic anonyms of $PA_i$ |
| $(pk, sk)$ | the pair of public and private keys |
| $E_{key}(M), D_{key}(M)$ | encrypt/decrypt the message $M$ by using key $key$ |
| $T^i$ | $ith$ timestamp |
| $h(\cdot)$ | a cryptographically secure one way hash function |
| $\oplus, \|\|$ | bitwise XOR operation and concatenation operation |

### A. GLOBAL SETUP PHASE

Trusted center $TC$ inputs a security parameter $\lambda$. Then, $TC$ runs global setup algorithm (as shown in Algorithm 1) to generate the system public parameter set $PP$ and system master secret key $MSK$. $PP$ is public in the system and $MSK$ is confidentially stored by $TC$. $PP$ is a default input in the following algorithm, which is omitted to simplify the presentation.

**Algorithm 1** Global Setup Algorithm

**Input:** parameter a security parameter $\lambda$.
**Output:** $(PP, MSK)$.

1: $TC$ chooses a $\lambda$-bit prime number $p$;
2: Selects a set of non-singular elliptic curve points $E(F_p)$ over prime field $F_p$;
3: Selects a $p$-order cyclic group $G_p$;
4: Selects $P$ as a generator of $G_p$ with order $n$;
5: Chooses two cryptographically secure one-way hash functions $h_1(\cdot)$: $\{0,1\}^* \rightarrow Z_p^*$, $h_2(\cdot)$: $\{0,1\}^* \rightarrow \{0,1\}^l$;
6: Chooses a cryptographic symmetric encryption/decryption pair $E_{key}(\cdot)/D_{key}(\cdot)$ with symmetric key $key$;
7: Chooses $\omega \in_R Z_p^*$;
8: Computes $Pub_{TC} = \omega P$;
9: Sets $PP = \{p, P, E_{key}(\cdot)/D_{key}(\cdot), Pub_{TC}, h_1(\cdot), h_2(\cdot)\}$;
10: Sets $MSK = \omega$;
11: **return** $(PP, MSK)$.

### B. PATIENT REGISTRATION

If a patient $PA_i$ wants to access medical care from the MediBchain system, he/she should register in trusted center $TC$ firstly. $PA_i$ chooses his/her $ID_{PA_i}$, secret value $x_{PA_i} \in_R Z_p^*$ and computes $P_{PA_i} = x_{PA_i}P$. Then, $PA_i$ sends registration request message $(ID_{PA_i}, P_{PA_i})$ to $TC$ through a secured channel. Upon receiving the registration message from $PA_i$, $TC$ runs patient registration algorithm (as shown in Algorithm 2) to generate $PA_i$'s static anonym $pid_{PA_i}$, partial secret/public key pair $(y_{PA_i}, Q_{PA_i})$. $TC$ generates $PA_i$'s static anonym $pid_{PA_i}$ using the master secret key $MSK$, a random value $\xi_{PA_i} \in_R Z_p^*$ and the symmetric encryption algorithm $E_{key}(\cdot)$(Line 1-2). $PA_i$'s partial secret/public key pair is constructed in Line 3-4.

**Algorithm 2** Patient Registration Algorithm

**Input:** $MSK$, $(ID_{PA_i}, P_{PA_i})$.
**Output:** $(pid_{PA_i}, (y_{PA_i}, Q_{PA_i}))$.

1: $TC$ chooses a random value $\xi_{PA_i} \in_R Z_p^*$;
2: Computes $pid_{PA_i} = E_\omega(ID_{PA_i}, \xi_{PA_i})$;
3: Chooses the other random value $q_{PA_i} \in_R Z_p^*$;
4: Computes $Q_{PA_i} = q_{PA_i}P$,
$\alpha_{PA_i} = h_1(P_{PA_i}||Q_{PA_i}||Pub_{TC})$,
$y_{PA_i} = \alpha_{PA_i}\omega + q_{PA_i}$;
5: **return** $(pid_{PA_i}, (y_{PA_i}, Q_{PA_i}))$.

According to Algorithm 2, $PA_i$ runs key generation algorithm (as shown in Algorithm 3) to set $(sk_{PA_i}, pk_{PA_i})$ as his/her own private/public key pair, where $sk_{PA_i} = (x_{PA_i}, y_{PA_i})$, $pk_{PA_i} = (P_{PA_i}, Q_{PA_i})$.

**Algorithm 3** $PA_i$'s Key Generation Algorithm

**Input:** $Pub_{TC}, x_{PA_i}, y_{PA_i}, P_{PA_i}, Q_{PA_i})$.
**Output:** $(sk_{PA_i}, pk_{PA_i})/\perp$.

$PA_i$ computes $\alpha_{PA_i} = h_1(P_{PA_i}||Q_{PA_i}||Pub_{TC})$;
**if** $y_{PA_i}P = \alpha_{PA_i}Pub_{TC} + Q_{PA_i}$ **then**
  $PA_i$ sets $sk_{PA_i} = (x_{PA_i}, y_{PA_i})$;
  Sets $pk_{PA_i} = (P_{PA_i}, Q_{PA_i})$;
**else**
  Output $\perp$;
**return** $(sk_{PA_i}, pk_{PA_i})/\perp$.

Thus, the registration of patient $PA_i$ has been completed.

### C. DOCTORS REGISTRATION

If a doctor $DO_k$ wants to be a legal medical care provider in the MediBchain network, similarly, he/she should register in trusted center $TC$ first. $DO_k$ with the identity $ID_{DO_k}$ chooses secret value $x_{DO_k} \in_R Z_p^*$ and computes $P_{DO_k} = x_{DO_k}P$. Then, $DO_k$ sends registration request message $P_{DO_k}$ to $TC$ through a public channel. Upon receiving the registration message from $DO_k$, $TC$ runs the doctors registration algorithm (as shown in Algorithm 4).

**Algorithm 4** Doctors Registration Algorithm

**Input:** $MSK$, $P_{DO_k}$.
**Output:** $(y_{DO_k}, Q_{DO_k})$.

1: $TC$ chooses a random value $q_{DO_k} \in_R Z_p^*$;
2: Computes $Q_{DO_k} = q_{DO_k}P$,
$\alpha_{DO_k} = h_1(P_{DO_k}||Q_{DO_k}||Pub_{TC})$,
$y_{DO_k} = \alpha_{DO_k}\omega + q_{DO_k}$;
3: **return** $(y_{DO_k}, Q_{DO_k})$.

According to Algorithm 4, $DO_k$ runs key generation algorithm (as shown in Algorithm 5) to set $(sk_{DO_k}, pk_{DO_k})$ as his/her own private/public key pair, where $sk_{DO_k} = (x_{DO_k}, y_{DO_k})$, $pk_{DO_k} = (P_{DO_k}, Q_{DO_k})$.

Thus, the registration of doctor $DO_k$ with the identity $ID_{DO_k}$ has been completed.

**Algorithm 5** $DO_k$'s Key Generation Algorithm

**Input:** $Pub_{TC}, x_{DO_k}, y_{DO_k}, P_{DO_k}, Q_{DO_k}$).
**Output:** $(sk_{DO_k}, pk_{DO_k})/\perp$.

1: $DO_k$ computes $\alpha_{DO_k} = h_1(P_{DO_k}||Q_{DO_k}||Pub_{TC})$;
2: **if** $y_{DO_k}P = \alpha_{DO_k}Pub_{TC} + Q_{DO_k}$ **then**
3: $DO_k$ sets $sk_{DO_k} = (x_{DO_k}, y_{DO_k})$;
4: Sets $pk_{DO_k} = (P_{DO_k}, Q_{DO_k})$;
5: **else**
6: Output $\perp$;
7: **return** $(sk_{DO_k}, pk_{DO_k})/\perp$.

### D. MEDIBCHAIN NODES REGISTRATION PHASE

In MBPA, if a MediBchain node wants to provide services to the terminals (patients and doctors), he/she should register in trusted center *TC* firstly to be a permissioned node (Same as Algorithms 4-5, the descriptions will not repeat again). Here, we put permissioned nodes in charge of maintaining the MediBchain using practical consensus mechanism (PBFT) [32], [33], which is unlike the Bitcoin blockchain (only used to proof the work, PoW). Like [33], the permissioned nodes here are also classified into two types: one is the ''validation node'', which is responsible for verifying transactions (abbreviated as *vdn*) and the other is the ''bookkeeping node'', which takes charge of chaining validated transactions into the MediBchain (abbreviated as *bkn*). After permissioned nodes reach a consensus on the received requests from terminals (patients and doctors), the requests will be recorded into the MediBchain. Here, we should declare that *bkn* only accepts transactions sent by its affiliated *vdn*.

The registration process of $vdn_j$ is similar to the one of $DO_k$. Eventually, $vdn_j$ obtains his/her private/public key pair $(sk_{vdn_j}, pk_{vdn_j})$, where $sk_{vdn_j} = (x_{vdn_j}, y_{vdn_j})$, $pk_{vdn_j} = (P_{vdn_j}, Q_{vdn_j})$ and $j = 1, 2, \ldots, t$. The cloud gateway's key pair is $(sk_C, pk_C)$, where $pk_C = sk_C P$

### E. MEDIBCHAIN NODE SHARED VALUE SETTING PHASE

Suppose $\{vdn_1, vdn_2, \ldots, vdn_t\}$ is the validation node set of MediBchain network. To guarantee the privacy preserving message delivery and minimize the number of the messages exchanged between the terminals and MediBchain nodes, *TC* runs MediBchain node secret shared value setting generation algorithm (shown in Algorithm 6) to generate a secret shared value key $\theta$ (Line 5), computes and releases $pk_{BC}$ as an extemporaneous group shared public key. The terminals encrypt the authentication messages by using the group shared public key $pk_{BC}$, which can be a huge time savings for message authentication. It is required that the authentication messages only can be authenticated by each node in $\{vdn_1, vdn_2, ..., vdn_t\}$ and the adversary cannot recover secret shared value $\theta$ from $pk_{BC}$.

### F. PATIENT REQUEST PHASE

Body sensor (in/on $PA_i$'s body) can measure $PA_i$'s health data $m_{PA_i} = (pid_{PA_i}, data_{PA_i}, T^1_{PA_i})$. $PA_i$ can obtain the data via his/her mobile device (such as smartphone, iPad) securely, make an appointment with the doctor $DO_k$ to get an appointment sequence number *sn*, and send *Transaction*1 to MediBchain network, which is stored in the cloud for later access by doctor $DO_k$ ultimately. Here, the ciphertext $C_{PA_i}$ in *Transaction*1 is encrypted by $key^1_{P-D}$, which is only shared between $PA_i$ and doctor $DO_k$. Hence, $C_{PA_i}$ can only be authenticated by $DO_k$ and the adversary cannot recover $m_{PA_i}$ from $C_{PA_i}$. The details of this phase are shown in Algorithm 7. After running Algorithm 7, $PA_i$ broadcasts this transaction to the MediBchain network.

**Algorithm 6** MediBchain Node Shared Value Setting Algorithm

**Input:** $(Pub_{TC}, pk_{vdn_j}(j = 1, 2, \ldots, t))$.
**Output:** $(a_0, a_1, \ldots, a_{t-1}, R, pk_{BC})$.

1: *TC* chooses a random value $r \in_R Z^*_p$;
2: Computes $R = rP$;
3: **for** $j = 1$ *to* $t$ **do**
4: Calculates $\alpha_{vdn_j} = h_1(P_{vdn_j}||Q_{vdn_j}||Pub_{TC})$, $S_{vdn_j} = r(\alpha_{vdn_j}Pub_{TC} + Q_{vdn_j} + P_{vdn_j})$;
5: Calculates $\beta_{vdn_j} = h_1(R, S_{vdn_j})$;
6: Chooses a random value $\theta \in_R Z^*_p$;
7: Computes $f(x) = \prod(x - \beta_{vdn_j}) + \theta \pmod p$ $= x^t + a_{t-1}x^{t-1} + \ldots + a_1 x + a_0$;
8: Computes $pk_{BC} = \theta P$;
9: **return** $(a_0, a_1, \ldots, a_{t-1}, R, pk_{BC})$.

**Algorithm 7** Patient Request Algorithm

**Input:** $(m_{PA_i}, sk_{PA_i}, pk_{PA_i}, pk_{BC}, ID_{DO_k}, pk_{DO_k}, sn)$.
**Output:** *Transaction*1.

1: $PA_i$ chooses a random value $a_{PA_i} \in_R Z^*_p$;
2: Computes $A_{PA_i} = a_{PA_i}P$,
$\alpha_{DO_k} = h_1(P_{DO_k}||Q_{DO_k}||Pub_{TC})$,
$key^1_{P-D}$
$= (x_{PA_i} + y_{PA_i} + a_{PA_i})(P_{DO_k} + \alpha_{DO_k}Pub_{TC} + Q_{DO_k})$,
$C_{PA_i}$
$= E_{key^1_{P-D}}(m_{PA_i}, h_2(m_{PA_i}, key^1_{P-D}), ID_{DO_k}, sn)$;
3: Computes $key_{P-B} = (x_{PA_i} + y_{PA_i} + a_{PA_i})pk_{BC}$,
$PID^1_{PA_i} = h_2(key^1_{P-D}, A_{PA_i}) \oplus pid_{PA_i}$,
$M_{i1} = h_2(PID^1_{PA_i}, C_{PA_i}, key_{P-B}, A_{PA_i}, pk_{PA_i}, pk_{BC})$,
$M_{i2} = h_2(pid_{PA_i}, ID_{DO_k}, C_{PA_i}, m_{PA_i}, key^1_{P-D}, A_{PA_i}, pk_{PA_i}, pk_{DO_k}, sn)$;
4: Sets *Transaction*1 $=$
$(PID^1_{PA_i}, ID_{DO_k}, A_{PA_i}, C_{PA_i}, M_{i1}, M_{i2}, sn)$;
5: **return** *Transaction*1.

### G. CHAIN TRANSACTION PHASE (I)

The messages in the MediBchain network are probably transmitted using the public channel, which are prone to be captured, tampered or forged by adversaries. In order to resist these attacks, it is important for $vdn_j$ to verify these encrypted messages. When receiving the

$Transaction1 = (PID^1_{PA_i}, ID_{DO_k}, A_{PA_i}, C_{PA_i}, M_{i1}, M_{i2}, sn)$ from $PA_i$, $vdn_j$ runs verify the transactions algorithm (shown in Algorithm 8) to authenticate $M_{i1}$. In Line 2, $vdn_j$ calculates the shared secret value $\theta$ to obtain the shared key $key_{P-B}$. $vdn_j$ verifies whether the following equation holds:

$$M_{i1} = h_2(PID^1_{PA_i}, C_{PA_i}, key_{P-B}, A_{PA_i}, pk_{PA_i}, pk_{BC})$$

If it holds (Line 4), it indicates that $Transaction1$ is a fresh effective and legal transaction sent by $PA_i$. Otherwise (Line 7), $Transaction1$ is rejected and outputs 0.

The phase of chaining transactions contains two steps. That is:

Step 1 For validating each transaction, $vdn_j$ first obtains invalid transactions and computers shared value $\theta$. Then, obtain $key_{P-B}$ to verify the transactions running Algorithm 8.

**Algorithm 8** Verify the Transactions Algorithm

**Input:** $((a_0, a_1, ..., a_{t-1}, R), sk_{vdn_j}, pk_{vdn_j}, pk_{PA_i}, Pub_{TC}, Transaction1)$.

**Output:** 1/0.

1: $vdn_j$ computes $\alpha_{PA_i} = h_1(P_{PA_i}||Q_{PA_i}||Pub_{TC})$,
$S_{vdn_j} = R(x_{vdn_j} + y_{vdn_j})$,
$\beta_{vdn_j} = h_1(R, S_{vdn_j})$;
2: Calculates $\theta = f(\beta_{vdn_j})$;
3: Computes
$key_{P-B} = \theta(P_{PA_i} + \alpha_{PA_i} Pub_{TA} + Q_{PA_i} + A_{PA_i})$;
4: **if** $M_{i1} = h_2(PID^1_{PA_i}, C_{PA_i}, key_{P-B}, A_{PA_i}, pk_{PA_i}, pk_{BC})$ **then**
5: **return** 1;
6: **else**
7: **return** 0.

Step 2 The $bkn_j$ utilizes the PBFT consensus mechanism for chaining the transactions. That is, the current leader repeatedly queries validated transactions (where the output of Algorithm 8 is 1) for a pending block. Then, a consensus is reached on this block, if and only if, at least two-third of total $bkn_j$ approve the pending block. Finally, the leader chains this block to the MediBchain.

### H. STORAGE AND RESPONSE PHASE(I)

Cloud gateway monitors the MediBchain network for new chained transactions. Once the terminal publishes a new valid request, the corresponding transaction generated and chained in the MediBchain will be received by the cloud. The cloud gateway is responsible for storing and replying data to the requester. He/she first stores the new chained transactions in the local cloud storage according to storage key $(PID^1_{PA_i}, A_{PA_i}, ID_{DO_k})$, which is for the convenience of terminal retrieval. Terminals can use either of them to retrieve stored transaction information. Next, the key $key_{P-C} = sk_C A_{PA_i}$ is generated, which can then be used to encrypt response data $(PID^1_{PA_i}, A_{PA_i}, ID_{DO_k})$. That is, the final response is $(E_{key_{P-C}}(PID^1_{PA_i}, A_{PA_i}, ID_{DO_k}), pk_C)$.

When $PA_i$ receives $(E_{key_{P-C}}(PID^1_{PA_i}, A_{PA_i}, ID_{DO_k}), pk_C)$, he/she first computes $key_{P-C} = a_{PA_i} pk_C$ using his/her one-time private key $a_{PA_i}$. Then, he/she recomputes $(PID^1_{PA_i}, A_{PA_i}, ID_{DO_k}) = D_{key_{P-C}}(PID^1_{PA_i}, A_{PA_i}, ID_{DO_k})$. If the equality holds, then it implies that the response is not from an impersonator. Hence, he/she obtains the response information about this request and believes that his/her request information has been saved to the cloud for later access by the doctor $DO_k$.

### I. TREATMENT PHASE

All information stored in the cloud is public and accessible to anyone without divulging terminal data. Because all the data is stored in the cloud as ciphertext. Only those who hold the corresponding secret key can decrypt the ciphertext. It undoubtedly saves doctors a lot of time in diagnosis process. Hence, the doctor $DO_k$ obtains $PA_i$'s data from the cloud without performing mutual authentication between $PA_i$ and cloud. After that, doctor $DO_k$ uses his/her identity $ID_{DO_k}$ to retrieve and obtain the stored transaction information $Transaction1$ in the cloud. $DO_k$ runs the treatment phase algorithm (shown in Algorithm 9) to recover the plaintext $m_{PA_i}$. If the both of two equations $h_2(m_{PA_i}, key^1_{P-D}) = h_2(m_{PA_i}, key^1_{P-D})$ and $M_{i2} = h_2(pid_{PA_i}, ID_{DO_k}, C_{PA_i}, m_{PA_i}, key^1_{P-D}, A_{PA_i}, pk_{PA_i}, pk_{DO_k}, sn)$ hold, $DO_k$ decrypts $C_{PA_i}$ and recovers $m_{PA_i}$ by utilizing his/her secret key $sk_{DO_k}$. According to the $m_{PA_i}$, the doctor $DO_k$ gives the corresponding diagnostic records $m_{DO_k}$ in the order of $sn$. Then, doctor $DO_k$ sends encrypted ciphertext $C_{DO_k}$ to MediBchain network, which will be stored in the cloud for later access by $PA_i$ timely. The details of this phase contain one algorithm as shown in Algorithm 9.

### J. CHAIN TRANSACTION PHASE (II)

When $vdn_j$ receives the $Transaction2$ from $DO_k$, $vdn_j$ obtains the shared value $\theta$ (Line 2) to calculate the encryption key $key_{D-B}$ (Line 3) and runs verify the transactions algorithm (shown in Algorithm 10). In Line 4, $vdn_j$ verifies whether the equation $M_{k1} = h_2(PID^2_{PA_i}, C_{DO_k}, key_{D-B}, A_{DO_k}, pk_{DO_k}, pk_{BC})$ holds, which is sent by $DO_k$. If it holds, it indicates that $Transaction2$ passes the verification and the Algorithm 10 outputs 1. The $bkn$ chains the $Transaction2$ for a pending block. Otherwise (Line 7), the Algorithm 10 outputs 0. The details of these operations are performed as shown in Algorithm 10:

Step 1 For validating each transaction, $vdn_j$ (Here, it is just an identifier of the authentication node, which may be the same as one in the previous process, and it may also be different.) first obtains invalid transactions and computers shared value $\theta$. Then, obtain $key_{D-B}$ to verify the transactions running Algorithm 8.

Step 2 The $bkn$ utilizes the PBFT consensus mechanism for chaining the transactions. That is, the current leader

**Algorithm 9** Treatment Phase Algorithm

**Input:** $Transaction1, sk_{DO_k}, pk_{DO_k}, ID_{DO_k}, pk_{PA_i}, Pub_{TC}$.

**Output:** $Transaction2/\perp$.

$ID_{DO_k}$ computes $\alpha_{PA_i} = h_1(P_{PA_i}||Q_{PA_i}||Pub_{TC})$;
Calculates
$key^1_{P-D} = (P_{PA_i} + \alpha_{PA_i} Pub_{TC} + Q_{PA_i} + A_{PA_i})(x_{DO_k} + y_{DO_k})$,
$pid_{PA_i} = PID^1_{PA_i} \oplus h_2(key^1_{P-D}, A_{PA_i})$;
Decrypts
$D_{key^1_{P-D}}(C_{PA_i}) = (m_{PA_i}, h_2(m_{PA_i}, key^1_{P-D})$
**if** $h_2(m_{PA_i}, key^1_{P-D}) = h_2(m_{PA_i}, key^1_{P-D})$ **then**
  **if** $M_{i2} = h_2(pid_{PA_i}, ID_{DO_k}, C_{PA_i}, m_{PA_i}, key^1_{P-D}, A_{PA_i}, pk_{PA_i}, pk_{DO_k}, sn)$ **then**
    $ID_{DO_k}$ chooses a random value $A_{DO_k} \in_R Z^*_p$;
    Computes $A_{DO_k} = a_{DO_k} P$,
    $key^2_{P-D} = (x_{DO_k} + y_{DO_k} + a_{DO_k})(P_{PA_i} + \alpha_{PA_i} Pub_{TC} + Q_{PA_i})$,
    $C_{DO_k} = E_{key^2_{P-D}}(m_{PA_i}, m_{DO_k}, h_2(m_{PA_i}, m_{DO_k}, key^2_{P-D}), ID_{DO_k}, pid_{PA_i}, sn)$;
    Computes $key_{D-B} = (x_{DO_k} + y_{DO_k} + A_{DO_k}) pk_{BC}$,
    $PID^2_{PA_i} = h_2(key^2_{P-D}, A_{DO_k}) \oplus pid_{PA_i}$,
    $M_{k1} = h_2(PID^2_{PA_i}, C_{DO_k}, key_{D-B}, A_{DO_k}, pk_{DO_k}, pk_{BC})$,
    $M_{k2} = h_2(pid_{PA_i}, ID_{DO_k}, C_{DO_k}, m_{DO_k}, key^2_{P-D}, A_{PA_i}, A_{DO_k}, pk_{PA_i}, pk_{DO_k}, sn)$;
    Sets $Transaction2 = (PID^2_{PA_i}, ID_{DO_k}, A_{DO_k}, C_{DO_k}, M_{k1}, M_{k2}, sn)$;
    **return** $Transaction2$.
  **else**
    **return** $\perp$.
**else**
  **return** $\perp$.

**Algorithm 10** Verify the Transactions Algorithm

**Input:** $((a_0, a_1, ..., a_{t-1}, R), sk_{vdn_j}, pk_{vdn_j}, pk_{DO_k}, Pub_{TA}, Transaction2)$.

**Output:** 1/0.

$vdn_j$ computes $\alpha_{vdn_j} = h_1(P_{vdn_j}||Q_{vdn_j}||Pub_{TC})$, $S_{vdn_j} = R(x_{vdn_j} + y_{vdn_j})$, $\beta_{vdn_j} = h_1(R, S_{vdn_j})$;
Calculates $\theta = f(\beta_{vdn_j})$;
Computes $key_{D-B} = \theta(P_{DO_k} + \alpha_{vdn_j} Pub_{TC} + Q_{DO_k} + A_{DO_k})$;
**if** $M_{k1} = h_2(PID^2_{PA_i}, C_{DO_k}, key_{D-B}, A_{DO_k}, pk_{DO_k}, pk_{BC})$ **then**
  **return** 1;
**else**
  **return** 0.

repeatedly queries validated transactions (where the output of Algorithm 10 is 1) for a pending block. Then, a consensus is reached on this block if, and only if, at least two-third of total *bkn*s approve the pending block. Finally, the leader chains this block to the MediBchain.

### K. STORAGE AND RESPONSE PHASE(II)

Cloud gateway monitors the MediBchain network for new chained transactions. Once a terminal publishes a new valid request, the corresponding transaction generated and chained in the MediBchain will be received by the cloud. The cloud gateway is responsible for storing and replying data to the requester. He/She first stores the new chained transactions in the local cloud storage according to storage key $(PID^2_{PA_i}, A_{DO_k}, ID_{DO_k})$, which is for the convenience of terminal retrieval. Terminals can use either of them to retrieve stored transaction information. Next, the key $key_{D-C} = sk_C A_{DO_k}$ is generated which can then be used to encrypt response data $(PID^2_{PA_i}, A_{DO_k}, ID_{DO_k})$, That is, the final response is $(E_{key_{D-C}}(PID^2_{PA_i}, A_{DO_k}, ID_{DO_k}), pk_C)$.

When the $DO_k$ receives $(E_{key_{D-C}}(PID^2_{PA_i}, A_{DO_k}, ID_{DO_k}), pk_C)$, it first computes the $key_{D-C} = a_{DO_k} pk_C$ using his/her one-time private key $A_{DO_k}$. Then, it recomputes $(PID^2_{PA_i}, A_{DO_k}, ID_{DO_k}) = D_{key_{D-C}}(PID^2_{PA_i}, A_{DO_k}, ID_{DO_k})$. If the equality holds, then it implies that the response is not from an impersonator. Hence, he/she obtains the response information about the this request.

**Algorithm 11** Checking up Phase Algorithm

**Input:** $(Transaction2, sk_{PA_i}, pk_{PA_i}, ID_{DO_k}, pk_{DO_k}, Pub_{TA})$.

**Output:** 1/0.

$PA_i$ computes $\alpha_{DO_k} = h_1(P_{DO_k}||Q_{DO_k}||Pub_{TC})$;
Calculates $key^2_{P-D} = (P_{DO_k} + \alpha_{DO_k} Pub_{TC} + Q_{DO_k} + A_{DO_k})(x_{PA_i} + y_{PA_i})$,
$pid_{PA_i} = PID^2_{PA_i} \oplus h_2(key^2_{P-D}, A_{DO_k})$;
Decrypts $D_{key^2_{P-D}}(C_{DO_k}) = (m_{PA_i}, m_{DO_k}, h_2(m_{PA_i}, m_{DO_k}, key^2_{P-D}))$
**if** $h_2(m_{PA_i}, m_{DO_k}, key^2_{P-D}) = h_2(m_{PA_i}, m_{DO_k}, key^2_{P-D})$ **then**
  **if** $M_{k2} = h_2(pid_{PA_i}, ID_{DO_k}, C_{DO_k}, m_{DO_k}, key^2_{P-D}, A_{PA_i}, A_{DO_k}, pk_{PA_i}, pk_{DO_k}, sn)$ **then**
    **return** 1.
  **else**
    **return** 0.
**else**
  **return** 0.

### L. CHECKING UP PHASE

After performing treatment from $DO_k$, the patient $PA_i$'s report is stored in the cloud. $PA_i$ uses $ID_{DO_k}$ to retrieve the stored transaction information $Transaction2$ in the cloud. $PA_i$ runs the checking up phase algorithm (shown in Algorithm 11) to recover the plaintext $m_{DO_k}$. $PA_i$ verifies whether the equations $h_2(m_{PA_i}, m_{DO_k}, key^2_{P-D}) = h_2(m_{PA_i}, m_{DO_k}, key^2_{P-D})$ and $M_{k2} = h_2(pid_{PA_i}, ID_{DO_k}, C_{DO_k}, m_{DO_k}, key^2_{P-D}, A_{PA_i}, A_{DO_k}, pk_{PA_i}, pk_{DO_k}, sn)$ hold. If they hold (The algorithm

outputs 1), $PA_i$ can obtains much better treatment based on the report. Otherwise (The Algorithm 11 outputs 0), $PA_i$ rejects the treatment report. The details of this phase are shown in Algorithm 11 and described below:

## V. SECURITY ANALYSIS AND PROOF OF MBPA SCHEME

In this section, we first make a security analysis to prove that the MBPA scheme can satisfy the security requirements aforementioned in the [1], [3], [33]. Second, we further prove that the MBPA scheme is provably secure based on the security model.

### A. SECURITY ANALYSIS

#### 1) COMPLETENESS AND MUTUAL AUTHENTICATION

In the MBPA scheme, all the authentication information $(M_{i1}, M_{i2}, M_{k1}, M_{k2})$ is based on secret values $(key_{P-B}, key_{D-B}, key^1_{P-D}, key^2_{P-D})$, where

$$\begin{aligned}
key_{P-B} &= (x_{PA_i} + y_{PA_i} + a_{PA_i})pk_{BC} \\
&= (x_{PA_i} + y_{PA_i} + a_{PA_i})\theta P \\
&= \theta(P_{PA_i} + \alpha_{PA_i}Pub_{TA} + Q_{PA_i} + A_{PA_i}); \\
key_{D-B} &= (x_{ID_k} + y_{ID_k} + a_{ID_k})pk_{BC} \\
&= (x_{ID_k} + y_{ID_k} + a_{ID_k})\theta P \\
&= \theta(P_{ID_k} + \alpha_{ID_k}Pub_{TA} + Q_{ID_k} + A_{ID_k}); \\
key^1_{P-D} &= (x_{PA_i} + y_{PA_i} + a_{PA_i})(P_{DO_k} + \alpha_{DO_k}Pub_{TC} + Q_{DO_k}) \\
&= (P_{PA_i} + \alpha_{PA_i}Pub_{TC} + Q_{PA_i} + A_{PA_i})(x_{DO_k} + y_{DO_k}); \\
key^2_{P-D} &= (P_{DO_k} + \alpha_{DO_k}Pub_{TC} + Q_{DO_k} + A_{DO_k})(x_{PA_i} + y_{PA_i}) \\
&= (x_{DO_k} + y_{DO_k} + a_{DO_k})(P_{PA_i} + \alpha_{PA_i}Pub_{TC} + Q_{PA_i})
\end{aligned}$$

(Here, $y_{PA_i} = \alpha_{PA_i}\omega + q_{PA_i}$, $y_{PA_i}P = \alpha_{PA_i}Pub_{TC} + Q_{PA_i}$ and $y_{DO_k} = \alpha_{DO_k}\omega + q_{DO_k}$, $y_{DO_k}P = \alpha_{DO_k}Pub_{TC} + Q_{DO_k}$)

$key_{P-B}$ is only shared between $PA_i$ and MediBchain, which anyone cannot obtain it except $PA_i$ and MediBchain. $key_{D-B}$ is only shared between $DO_k$ and MediBchain, which anyone cannot obtain it except $DO_k$ and MediBchain. $key^1_{P-D}$ is only shared between $PA_i$ and $DO_k$, which anyone cannot obtain it except $PA_i$ and $DO_k$. In the whole scheme as shown in Algorithms 1-11, MediBchain nodes can authenticate a terminal (patient/doctor) based on the validation of their published transactions. In addition, the terminal can identify a legitimate cloud by recomputing the $key_{P-C}/key_{D-C}$. Any probabilistic polynomial time adversary cannot successfully forge a valid $key_{P-C}/key_{D-C}$ for a target message because solving ECDHP is computationally hard. Clearly, MBPA scheme achieves mutual authentication between terminals and MediBchain, terminals ($PA_i$) and terminals ($DO_k$).

#### 2) PATIENT ANONYMITY

The MBPA scheme adopts the anonymous blind identities $PID^1_{PA_i} = h_2(key^1_{P-D}, A_{PA_i}) \oplus pid_{PA_i}$ and $PID^2_{PA_i} = h_2(key^2_{P-D}, A_{DO_k}) \oplus pid_{PA_i}$ instead of the static anonym $pid_{PA_i}$ in the public communication channel. Meanwhile, they are different in each run. In addition to $PA_i$'s static anonym $pid_{PA_i}$, the doctor $DO_k$ cannot obtain anything about the $PA_i$'s true identity $ID_i$. Here, $pid_{PA_i} = E_\omega(ID_{PA_i}, \xi_{PA_i})$. By using a secure cryptographic symmetric encryption, the malicious adversary $\mathcal{A}$ cannot extract the $ID_i$ without knowing $\omega$ required to successfully decrypt the ciphertext, further. In this way, the MBPA scheme provides patient anonymity, which can prevent the privacy leakage of patient.

#### 3) PATIENT TRACEABILITY

If a patient $PA_i$ sends some false messages to deceive doctors, $TC$ can extract real identity of $PA_i$ by decrypting $pid_{PA_i}$ using his/her private key $\omega$. Hence, the MBPA scheme achieves patient traceability to prevent malicious patients from doing something to harm systems.

#### 4) PERFECT FORWARD SECRECY

In MBPA scheme, even after an attacker $\mathcal{A}$ has successfully compromised both public/private key pairs of a terminal (patient/doctor) and MediBchain nodes, he/she can only reveal the ciphertext belonging to the current session. Since the ephemeral key of a terminal (patient/doctor) is different in each session, perfect forward secrecy is guaranteed.

#### 5) NO VERIFIER TABLE

As described above, mutual authentication only requires the terminal (patient/doctor), MediBchain nodes and cloud to generate public/private key pairs without the help of $TC$. Then, these keys are used to generate ciphertext and messages for authentication. Clearly, there is no verifier table maintained by $TC$.

#### 6) BIRTHDAY COLLISION RESILIENCE

Similar to [33], MBPA scheme also has the such nature. Here, we also utilize the PBFT consensus mechanism in the permissioned MediBchain to chain a block. The permissioned participate MediBchain nodes are responsible for recording and adding the valid transactions into a pending block. These blocks reach consensus based on shared secrets. Hence, there is no fork situation, which effectively avoids blocks' birthday collisions.

#### 7) RESILIENCE TO IMPERSONATION ATTACK

If $\mathcal{A}$ can obtain the information $Transaction1 = (PID^1_{PA_i}, ID_{DO_k}, A_{PA_i}, C_{PA_i}, M_{i1}, M_{i2}, sn)$ and $Transaction2 = (PID^2_{PA_i}, ID_{DO_k}, A_{DO_k}, C_{DO_k}, M_{k1}, M_{k2}, sn)$ in public channel. Here, $M_{i1}$ is only shared between $PA_i$ and MediBchain nodes. $M_{k1}$ is only shared between $DO_k$ and MediBchain nodes. $M_{i2}$ and $M_{k2}$ are only shared between $PA_i$ and $DO_k$. So $\mathcal{A}$ can not figure out the valid authentication messages $(M_{i1}, M_{i2}, M_{k1}, M_{k2})$ to pass the authentication. Hence, the MBPA scheme can resist the impersonation attack.

#### 8) RESILIENCE TO INTERNAL ATTACKS

Assume that $\mathcal{A}$ is a malicious-legitimate patient(doctor), $\mathcal{A}$ uses his/her own information in public channel. He/She

obtains nothing about other patients'(doctors') secret information $key_{P-B}$, $key_{D-B}$, $key^1_{P-D}$ and $key^2_{P-D}$. He/She also cannot get the random values $a_{PA_i}$ or $a_{DO_k}$. So, he/she cannot succeed in forging authentication information $(M_{i1}, M_{i2}, M_{k1}, M_{k2})$ to pass the authentication. Hence, the MBPA scheme can resist the internal attacks.

#### 9) RESILIENCE TO REPLAY ATTACK

Suppose $\mathcal{A}$ intercepts the *Transaction*1 and *Transaction*2, and replies these messages to corresponding partners. The communicating parties (i.e. $PA_i$, $DO_k$) generate new random numbers $(a_{PA_i}, a_{DO_k})$ in each session, which are involved in patients request authenticator and doctors response authenticator. So that the patient, MediBchain nodes or the doctor can check the freshness of those random numbers. What's more, the freshness of these random numbers can be determined by the transaction details on the MediBchain. Therefore, the MBPA scheme resists replay attack.

#### 10) RESILIENCE TO MAN-IN-THE-MIDDLE ATTACK

In this attack, $\mathcal{A}$ may try to impersonate a valid patient $PA_i$, or his partner $DO_k$ by intercepting the message. However, in the MBPA scheme the secret values $key^1_{P-D}$ and $key^2_{P-D}$ are only shared between $PA_i$ and $DO_k$, they will never be discovered by anybody else except $PA_i$ and $DO_k$. Hence, the MBPA scheme is secure against man-in-the-middle attack.

### B. SECURITY MODEL

There are two types adversaries who have different abilities considered in certificateless cryptography: type-1 $\mathcal{A}_1$ and type-II $\mathcal{A}_2$ [34]–[37]. $\mathcal{A}_1$ cannot access the systems master key, but it can replace any users public key. $\mathcal{A}_2$ cannot replace users public key, but it can access the systems master key. The security model of MBPA scheme is defined by the following two games between a challenger $\mathcal{C}$ who wants to solve the ECDHP and an adversary $\mathcal{A}_i(i = 1, 2)$ who wants to forge a legitimate login/response message. Let $U$ denote an instance of a participant $\Omega$, where $\Omega$ is a patient or a doctor.

*Game-1:* Game-1 is interactive between $\mathcal{C}$ and $\mathcal{A}_1$.

*Setup:* Given a secure parameter $\lambda$, $\mathcal{C}$ generates the system parameters $PP$ and the master key $\omega$. Finally, $\mathcal{C}$ returns $PP$. The adversary $\mathcal{A}_1$ is allowed to ask queries to the following oracles:

- *Hash query:* $\mathcal{A}_1$ is allowed to issue all hash oracles and can obtain the corresponding hash value.
- *Symmetric encryption query:* $\mathcal{A}_1$ is allowed to issue all symmetric encryption query and can obtain the corresponding ciphertext.
- *Extract ephemeral key reveal query:* Upon receiving $\mathcal{A}_1$'s query, $\mathcal{C}$ outputs the corresponding ephemeral key to $\mathcal{A}_1$.
- *Extract secret value query of (U):* Upon receiving $\mathcal{A}_1$'s query, $\mathcal{C}$ outputs the corresponding secret value to $\mathcal{A}_1$.
- *Extract partial secret key query of (U):* Given a user with identity $ID_U$, $\mathcal{C}$ computes the corresponding partial private key $q_{ID_U}$ and transmits $q_{ID_U}$ to $\mathcal{A}_1$.
- *Request public key query of (U):* $\mathcal{A}_1$ can request any users public key.
- *Replace public key query of (U):* $\mathcal{A}_1$ can replace any users public key.
- *Send query of (U, M):* Upon receiving $\mathcal{A}_1$'s query, $\mathcal{C}$ should generate a respond information for the received message $M$ to $\mathcal{A}_1$.
- *Reveal query of (U):* Upon receiving the query, $\mathcal{C}$ returns the session key between $U$ and its partner to $\mathcal{A}_1$, if $\mathcal{C}$ has accepted. Otherwise, $\mathcal{C}$ outputs $\perp$ to $\mathcal{A}_1$.
- *Corrupt query of (U):* Obtaining the corrupt query, $\mathcal{C}$ returns $U$'s secret key to $\mathcal{A}_1$.

*Game-2:* Game-2 is interactive between $\mathcal{C}$ and $\mathcal{A}_2$.

*Setup:* Given a secure parameter $\lambda$, $\mathcal{C}$ generates systems parameters $PP$ and the systems master key $\omega$. In addition, $\mathcal{C}$ generates $Q_{PA_i}$ and $Q_{DO_k}$ as a part of the senders public key and the receivers public key, respectively. Finally, $\mathcal{C}$ outputs $PP$, $Pub_{TC}$, $Q_{PA_i}$ and $Q_{DO_k}$. $\mathcal{A}_2$ is allowed to ask Hash query, Symmetric encryption query, Extract partial secret key query, Send query, Reveal query and Corrupt query as in Game-1.

### C. SECURITY PROOF

Assuming that the ECDHP is hard, the security of the MBPA is demonstrated blow.

*Theorem 1:* In the random oracle, if there exists a type-1 adversary $\mathcal{A}_1$, who is able to forge a legitimate treatment message or its partner's respond message with a non-negligible probability $\epsilon$ in time $T$. We show that there is a challenger $\mathcal{C}$ who can solve the ECDHP with a non-negligible probability

$$\epsilon' \geq (1 - \frac{2}{q_{ep}+1})^{q_{ep}}(1 - \frac{2}{q_s+1})^{q_s}\frac{1}{nm}\frac{2}{q_{se}}\frac{4}{q_{h_2}}\epsilon.$$

in time

$$t' \leq T + q_{se}(2t_{se} + t_{sd}) + 4(q_{es} + q_{ep} + q_s)t_{sm},$$

where $q_{se}, q_{h_i}, q_{es}, q_{ep}, q_s$ denote the times of symmetric-encryption queries, hash-query, extract-secret-value queries, extract-partial-secret-value queries and send queries. $n$ and $m$ denote the number of patients and doctors separately. Let $t_{se}$, $t_{sd}$ and $t_{sm}$ denote the time of symmetric-encryption, symmetric-decryption and scalar multiplications separately.

*Proof:* Let $\mathcal{C}$ be a ECDHP challenger who receives a random instance ($P$,$Q_1 = aP$,$Q_2 = bP$) of ECDHP in $G_p$. A type-1 adversary $\mathcal{A}_1$ interacts with $\mathcal{C}$ as follows. We show how $\mathcal{C}$ may use $\mathcal{A}_1$ to solve the ECDHP, that is to compute $abP$.

*Setup:* $\mathcal{C}$ randomly selects a patient $PA_I$ as the challenge patient and his/her responder doctor $ID_K$ as the challenge doctor. Then, $\mathcal{C}$ generates three numbers $\alpha_{PA_I}, x_{PA_I}, q_{PA_I} \in Z_p^*$ randomly, computes $Pub_{TC} = \alpha^{-1}_{PA_I}(Q_1 - q_{PA_I}P - x_{PA_I}P)$ and gives $\{p, P, G_p, Pub_{TC}, h_1(\cdot), h_2(\cdot)\}$ to $\mathcal{A}_1$ as public parameters. $\mathcal{C}$ maintains the following lists to avoid inconsistency and for a quick response to the adversary $\mathcal{A}_1$:

**Symmetric encryption query:** $\mathcal{C}$ maintains a list $L_{se}$, which contains tuples $(m_k, key_k, c_k)$. Upon receiving $\mathcal{A}_1$'s query on $(m_k, key_k, c_k)$, if the tuple $(m_k, key_k, c_k)$ already in

the $L_{se}$ list, $\mathcal{C}$ outputs $c_k$. Otherwise, $\mathcal{C}$ selects a randomized string $c_k \in \{0,1\}^*$, adds $(m_k, key_k, c_k)$ to the $L_{se}$ list and returns $c_k$ to $\mathcal{A}_1$.

**Hash query:** $\mathcal{C}$ maintains a list $L_{h_k}$ of tuples $(m_k, n_k)$ initialized empty. Upon receiving $\mathcal{A}_1$'s query on $(m_k, n_k)$, $\mathcal{C}$ checks whether the tuple $(m_k, n_k)$ exists in $L_{h_k}$. If so, $\mathcal{C}$ returns $n_k$ to $\mathcal{A}_1$ directly. Otherwise, $\mathcal{C}$ chooses a random number $n_k \in Z_p^*$ or a random string $n_k \in \{0,1\}^l$ back to $\mathcal{A}_1$ and adds the tuple $(m_k, n_k)$ into the $L_{h_k}$, where $k = 1, 2$.

**Extract ephemeral key reveal query of $(U)$:** $\mathcal{C}$ maintains a list $L_U^1$ containing tuples $(U, a_U, A_U)$ initialized empty. Upon receiving $\mathcal{A}_1$'s query on $(U, a_U, A_U)$. $\mathcal{C}$ checks whether a tuple $(U, a_U, A_U)$ exists in $L_U^1$. If it exists, $A_U$ is returned directly. Otherwise, $\mathcal{C}$ selects a random number $a_U \in Z_p^*$, computes $A_U = a_U P$, adds the new tuple $(U, a_U, A_U)$ in $L_U^1$ and sends $A_U$ to $\mathcal{A}_1$.

**Extract secret value query of $(U)$:** $\mathcal{C}$ maintains a list $L_U^2$ containing tuples $(U, ID_U, x_U, P_U)$ initialized empty. Upon receiving $\mathcal{A}_1$'s query on $(U, ID_U, x_U, P_U)$. $\mathcal{C}$ checks whether a tuple $(U, ID_U, x_U, P_U)$ exists in $L_U^2$. If it exists, $x_U$ is returned directly. Otherwise, $\mathcal{C}$ selects a random number $x_U \in Z_p^*$, computes $P_U = x_U P$, adds the new tuple $(U, ID_U, x_U, P_U)$ in $L_U^2$ and sends $x_U$ to $\mathcal{A}_1$.

**Extract partial secret key query of $(U)$:** $\mathcal{C}$ maintains several initialized-empty lists $L_U^3$. Upon receiving the partial secret key query on the user $U$, $\mathcal{C}$ checks whether a tuple $(U, Q_U, y_U)$ exists in $L_U^3$. If so, $y_U$ is returned directly. Otherwise, $\mathcal{C}$ calculates as following:

• If $U = PA_I$, $\mathcal{C}$ selects random number $q_{PA_I}, \alpha_{PA_I} \in Z_p^*$, computes $Q_{PA_I} = q_{PA_I} P$, sets $y_{PA_I} = \perp$ and reads $P_{PA_I}$ from the list $L_{PA_I}^1$ according to $PA_I$. Then, $\mathcal{C}$ computes $\alpha_{PA_I} = h_1(P_{PA_I} || Q_{PA_I} || Pub_{TC})$. At last, $\mathcal{C}$ stores $(PA_I, P_{PA_I}, Q_{PA_I}, Pub_{TC}, \alpha_{PA_I})$ and $(PA_I, Q_{PA_I}, \perp)$ into $L_{h_1}$ and $L_U^3$ separately. • If $U = DO_K$, $\mathcal{C}$ selects random number $\alpha_{DO_K} \in Z_p^*$, reads $A_{DO_K}, P_{DO_K}$ from the list $L_U^1$ and $L_U^2$ according to $DO_K$, computes $Q_{DO_K} = Q_2 - \alpha_{DO_K} Pub_{TC} - P_{DO_K} - A_{DO_K}$, sets $y_{DO_K} = \perp$. At last, $\mathcal{C}$ adds $(DO_K, P_{DO_K}, Q_{DO_K}, Pub_{TC}, \alpha_{DO_K})$ and $(DO_K, Q_{DO_K}, \perp)$ into $L_{h_1}$ and $L_U^3$ separately. • Otherwise, $\mathcal{C}$ generates $(U, Q_U, y_U)$ using the user (patient/doctor) registration algorithm in the MBPA scheme. The tuple $(U, P_U, Q_U, Pub_{TC}, \alpha_U)$ is inserted into $L_{h_1}$, and the tuple $(U, Q_U, y_U)$ is inserted into $L_U^3$ separately.

**Request public key query of $(U)$:** An initialized-empty list $L_U^4$ is utilized to store the query result. Obtaining a request public key on user $U$, $\mathcal{C}$ checks whether a tuple $(U, x_U, P_U, q_U, Q_U)$ exists in $L_U^4$. If it exists, $(U, P_U, Q_U)$ is returned directly. Otherwise, $\mathcal{C}$ responds $(U, P_U, Q_U)$ by accessing to list $L_U^2$ and list $L_U^3$ and set $d_U := 0$ ($d_U$ denotes the time of public key replacement). At last, the tuple $(U, x_U, P_U, q_U, Q_U, d_U)$ is inserted to $L_U^4$.

**Replace public key query of $(U)$:** Upon receiving the replace public key query on the user $U$, $\mathcal{C}$ first makes a request public key on $U$ and finds the tuple $(U, x_U, P_U, q_U, Q_U, d_U)$ on $L_U^3$. Then, $\mathcal{C}$ replaces $pk_U = (P_U, Q_U)$ with $pk'_U = (P'_U, Q'_U)$ which is chosen by $\mathcal{A}_1$ and puts $d_U := d_U + 1$. At last, the tuple $(U, x'_U, P'_U, q'_U, Q'_U, d_U)$ is inserted to $L_U^5$.

**Send query of $(U, M)$:** Upon receiving the query with message $M$, C answers the query as follows. Obtaining the send query with mesage $M$, $\mathcal{C}$ responds the query as follows:

• $M = Transaction1$: The query is message $M$ from $PA_i$ to $DO_k$.

• If $PA_i = PA_I$, $\mathcal{C}$ aborts the session. • If $PA_i \neq PA_I$, $DO_k = DO_K$, $\mathcal{C}$ aborts the session. • If $PA_i \neq PA_I$, $DO_k \neq DO_K$, $\mathcal{C}$ runs according to the specification of the protocol, where $\mathcal{C}$ knows the private key of $PA_i$.

• $M = Transaction2$: The query is message $M$ from $DO_k$ to $PA_i$.

• If $DO_k = DO_K$, $\mathcal{C}$ aborts the session. • If $DO_k \neq DO_K$, $PA_i = PA_I$, $\mathcal{C}$ aborts the session. • If $DO_k \neq DO_K$, $PA_i \neq PA_I$, $\mathcal{C}$ runs according to the specification of the protocol, where $\mathcal{C}$ knows the private key of $DO_k$.

**Reveal query of $(U)$:** Upon receiving the query, $\mathcal{C}$ checks if $U = PA_I$ or $U = DO_K$. If yes, $\mathcal{C}$ aborts the session. Otherwise, $\mathcal{C}$ returns the shared key between $U$ and its partner to $\mathcal{A}_1$.

**Corrupt query of $(U)$:** Obtaining the corrupt query, $\mathcal{C}$ checks the list $L_U^1$, $L_U^2$ and the list $L_U^3$ for the tuples $(U, a_U, A_U)$, $(U, ID_U, x_U, P_U)$ and $(U, Q_U, y_U)$. Then, $\mathcal{C}$ returns $(U, a_U, A_U, x_U, P_U, Q_U, y_U)$ to $\mathcal{A}_1$.

Finally, $\mathcal{A}_1$ outputs a legitimate authentication message $(M_{k1}, M_{k2})$. If $(PK_i, DO_k) \neq (PK_I, DO_K)$, $\mathcal{C}$ aborts the game. Otherwise, $\mathcal{C}$ randomly chooses a tuple $(*, key_{P-D}^2, *)$ or $(*, M_{k2}, *)$ from the list $L_{se}$ and $L_{h_2}$ and outputs $key_{P-D}^2$ or $M_{k2}$ as the solution of ECDHP.

To complete the proof, we shall show that $\mathcal{C}$ solves the given instance of ECDHP with probability $\epsilon'$. First, we analyze several events for $\mathcal{C}$ to succeed:

$E1$: $\mathcal{C}$ does not abort any $\mathcal{A}_1$'s "Extract partial secret key queries".
$E2$: $\mathcal{C}$ does not abort any $\mathcal{A}_1$'s "Send queries".
$E3$: $\mathcal{C}$ obtains a legitimate authentication message $M_{k2}$.
$E4$: $(PA_i, DO_k) = (PA_I, DO_K)$.
$E5$: $\mathcal{C}$ chooses a correct tuple from the list $L_{h_2}$.
$E6$: $\mathcal{C}$ chooses a correct tuple from the list $L_{se}$.

Then, we have:

$$\Pr[E1] \geq (1 - \frac{2}{q_{ep}+1})^{q_{ep}}$$
$$\Pr[E2|E1] \geq (1 - \frac{2}{q_s+1})^{q_s}$$
$$\Pr[E3|E1 \wedge E2] \geq \epsilon$$
$$\Pr[E4|E1 \wedge E2 \wedge E3] \geq \frac{1}{nm}$$
$$\Pr[E5|E1 \wedge E2 \wedge E3 \wedge E4] \geq \frac{4}{q_{h_2}}$$
$$\Pr[E6|E1 \wedge E2 \wedge E3 \wedge E4 \wedge E5] \geq \frac{2}{q_{se}}$$

Hence, we have:
$\epsilon' = \Pr[E1 \wedge E2 \wedge E3 \wedge E4 \wedge E5 \wedge E6] = \Pr[E1]\Pr[E2|E1]\Pr[E3|E1 \wedge E2]\Pr[E4|E1 \wedge E2 \wedge E3]\Pr[E5|E1 \wedge E2 \wedge E3 \wedge E3]\Pr[E6|E1 \wedge E2 \wedge E3 \wedge E4 \wedge E5] \geq (1 - \frac{2}{q_{ep}+1})^{q_{ep}}(1 - \frac{2}{q_s+1})^{q_s}\frac{1}{nm}\frac{2}{q_{se}}\frac{4}{q_{h_2}}\epsilon$.

The running time $t$ for $\mathcal{C}$ is the sum of $\mathcal{A}_1$'s running time, the time that $\mathcal{C}$ responds queries and the time that $\mathcal{C}$ computes the ECDHP. Hence,

$$t' \leq T + q_{se}(2t_{se} + t_{sd}) + 4(q_{es} + q_{ep} + q_s)t_{sm}.$$

*Theorem 2:* In the random oracle, if there exists a type-2 adversary $\mathcal{A}_2$, who is able to forge a legitimate treatment message or its partner's respond message with a non-negligible probability $\epsilon$ in time $T$. We show that there is a challenger $\mathcal{C}$, who can solve the ECDHP with a non-negligible probability $\epsilon'$, where

$$\epsilon' \geq (1 - \frac{2}{q_{es}+1})^{q_{es}}(1 - \frac{2}{q_{sq}+1})^{q_{sq}}\frac{1}{nm}\frac{2}{q_{H_3}}\epsilon.$$

in time

$$t' \leq T + 2q_{se}t_{se} + q_{se}t_{sd} + 2(q_{es} + 2q_{sq})t_{sm}.$$

*Proof:* Let $\mathcal{C}$ be a ECDHP challenger who receives a random instance ($P$,$Q_1 = aP$,$Q_2 = bP$) of ECDHP in $G_p$. A type-2 adversary $\mathcal{A}_2$ interacts with $\mathcal{C}$ as follows. We show how $\mathcal{C}$ may use $\mathcal{A}_2$ to solve the ECDHP, that is to compute $abP$.

*Setup:* $\mathcal{C}$ randomly selects a patient $PA_I$ as challenge patient and his/her responder doctor $ID_K$ as the challenge doctor. Then, $\mathcal{C}$ generates a number $\omega \in Z_p^*$ randomly, computes $Pub_{TA} = \omega P$ and gives $\{p, P, G_p, Pub_{TA}, h_1(\cdot), h_2(\cdot)\}$ to $\mathcal{A}_2$ as public parameters. $\mathcal{C}$ maintains the following lists to avoid inconsistency and for a quick response to the adversary $\mathcal{A}_2$:

Due to the initiate-respond process of "**Symmetric encryption query**", "**Hash query**", "**Extract ephemeral key reveal query**" and "**Extract secret value query**" are same as what do in **Theorem 1.**. We will not repeat them here. For more details, please refer to Theorem 1.

**Request public key of $(U)$:** An initialized-empty list $L_U^4$ is utilized to store the query result. Obtaining a request public key of $U$. $\mathcal{C}$ checks whether a tuple $(U, x_U, P_U, q_U, Q_U)$ exists in $L_U^4$. If it exists, $(U, P_U, Q_U)$ is returned. Otherwise, $\mathcal{C}$ calculates as following:

• If $U = PA_I$, $\mathcal{C}$ obtains $\alpha_{PA_I}, P_{PA_I}$ by accessing to $L_{h_1}$ and $L_{PA_I}^2$, and computes $Q_{PA_I} = Q_1 - \alpha_{PA_I}Pub_{TA} - P_{PA_I}$. At last, the tuple $(PA_I, P_{PA_I}, Q_{PA_I})$ is inserted to $L_{PA}^4$. • If $U = DO_K$, $\mathcal{C}$ obtains $\alpha_{DO_K}, P_{DO_K}, A_{DO_K}$ by accessing to $L_{h_1}, L_{DO_K}^1$ and $L_{DO_K}^2$, and computes $Q_{DO_K} = Q_2 - \alpha_{DO_K}Pub_{TA} - P_{DO_K} - A_{DO_K}$. At last, the tuple $(DO_K, P_{DO_K}, Q_{DO_K})$ is inserted to $L_{DO_K}^4$. • Otherwise, $\mathcal{C}$ generates $(U, P_U, Q_U)$ using the user (patient/doctor) registration algorithm in the MBPA scheme. The tuple $(U, P_U, Q_U, Pub_{TC}, \alpha_U)$ is inserted into $L_{h_1}$, and the tuple $(U, Q_U, y_U)$ is inserted into $L_U^3$ separately.

**Send query of $(U, M)$:** Obtaining the send query with message $M$, $\mathcal{C}$ responds the query as follows:

$M$ = *Transaction*1: The query is message $M$ from $PA_i$ to $DO_k$.
• If $PA_i = PA_I$, $\mathcal{C}$ aborts the session.
• If $PA_i \neq PA_I$, $DO_k = DO_K$, $\mathcal{C}$ aborts the session.
• If $PA_i \neq PA_I$, $DO_k \neq DO_K$, $\mathcal{C}$ runs according to the specification of the protocol, where $\mathcal{C}$ knows the private key of $PA_i$.

$M$ = *Transaction*2: The query is message $M$ from $DO_k$ to $PA_i$.
• If $DO_k = DO_K$, $\mathcal{C}$ aborts the session.
• If $DO_k \neq DO_K$, $PA_i = PA_I$, $\mathcal{C}$ aborts the session.
• If $DO_k \neq DO_K$, $PA_i \neq PA_I$, $\mathcal{C}$ runs according to the specification of the protocol, where $\mathcal{C}$ knows the private key of $DO_k$.

**Reveal query of $(U)$:** Upon receiving the query, $\mathcal{C}$ checks if $U = PA_I$ or $U = DO_K$. If so, $\mathcal{C}$ aborts the session. Otherwise, $\mathcal{C}$ returns the shared key between $U$ and its partner to $\mathcal{A}_2$.

**Corrupt query of $(U)$:** Obtaining the corrupt query, $\mathcal{C}$ checks the list $L_U^1$, $L_U^2$ and the list $L_U^3$ for the tuples $(U, a_U, A_U)$, $(U, ID_U, x_U, P_U)$ and $(U, Q_U, y_U)$. Then, $\mathcal{C}$ returns $(U, a_U, A_U, x_U, P_U, Q_U, y_U)$ to $\mathcal{A}_2$.

To complete the proof, we shall show that $\mathcal{C}$ solves the given instance of ECDHP with probability $\epsilon'$. First, we analyze several events for $\mathcal{C}$ to succeed:

$E1$: $\mathcal{C}$ does not abort any $\mathcal{A}_2$'s "Extract secret value queries".
$E2$: $\mathcal{C}$ does not abort any $\mathcal{A}_2$'s "Send queries".
$E3$: $\mathcal{C}$ obtains a legitimate authentication message $M_{k2}$.
$E4$: $(PA_i, DO_k) = (PA_I, DO_K)$.
$E5$: $\mathcal{C}$ chooses a correct tuple from the list $L_{h_2}$.
$E6$: $\mathcal{C}$ chooses a correct tuple from the list $L_{se}$.
Then, we have:

$$\Pr[E1] \geq (1 - \frac{2}{q_{es}+1})^{q_{es}}$$

$$\Pr[E2|E1] \geq (1 - \frac{2}{q_s+1})^{q_s}$$

$$\Pr[E3|E1 \wedge E2] \geq \epsilon$$

$$\Pr[E4|E1 \wedge E2 \wedge E3] \geq \frac{1}{nm}$$

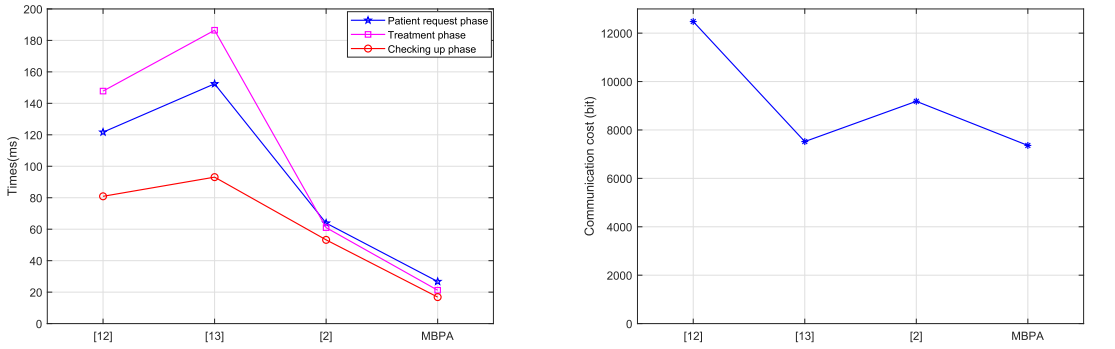
$$\Pr[E5|E1 \wedge E2 \wedge E3 \wedge E4] \geq \frac{4}{q_{h_2}}$$

$$\Pr[E6|E1 \wedge E2 \wedge E3 \wedge E4 \wedge E5] \geq \frac{2}{q_{se}}$$

Hence, we have:
$\epsilon' = \Pr[E1 \wedge E2 \wedge E3 \wedge E4 \wedge E5] = \Pr[E1]\Pr[E2|E1]\Pr[E3|E1 \wedge E2]\Pr[E4|E1 \wedge E2 \wedge E3]\Pr[E5|E1 \wedge E2 \wedge E3 \wedge E3] \geq (1 - \frac{2}{q_{es}+1})^{q_{es}}(1 - \frac{2}{q_s+1})^{q_s}\frac{1}{nm}\frac{4}{q_{h_2}}\frac{2}{q_{se}}\epsilon$.

The running time $t$ for $\mathcal{C}$ is the sum of $\mathcal{A}_2$'s running time, the time that $\mathcal{C}$ responds queries and the time that $\mathcal{C}$ computes the ECDHP. Hence,

$$t' \leq T + q_{se}(2t_{se} + t_{sd}) + 4(q_{es} + q_s)t_{sm}.$$




(a) Computational times comparisons in terminals' side (b) Communication cost comparisons in the three phases

**FIGURE 4.** **Performance comparisons of related medical cloud authentication schemes.**

## VI. PERFORMANCE ANALYSIS AND COMPARISON

In this section, we present the rigorous performance analysis and comparison of MBPA compared with other existing schemes [2], [12], [13] from the computation cost, the communication cost and security features.

### A. COMPUTATION COST

For the sake of discussion, let $t_h$, $t_c$, $t_x$, $t_{bp}$, $t_{sm}$, $t_{sg}$, $t_{sv}$, $t_{se}$, $t_{sd}$ and $t_{hp}$ denote the execution timings of hash function of SHA-256, concatenation operation, XOR operation, one paring operation, scalar multiplication operation, ABS signature generation operation, ABS signature verification operation, symmetric encryption operation, symmetric decryption operation and hash-to-point operation. Since $t_h$, $t_c$ and $t_x$ are negligible as compared to the other seven operations, we do not take them into account [38], [39].

Here, we mainly focus on the efficiency of three algorithms in terminals' side (patients and doctors): patient request algorithm, treatment phase algorithm and checking up phase algorithm. Since these three algorithms are the main body of MBPA and are executed much more frequently than the other phases. The simulations of these algorithms are implemented at Java using an Intel(R) Core(TM) i5-7000 CPU@ 3.40GHZ with 7.9GB RAM in Ubuntu 16.04 system. We run each operation in multiple of tens and then calculate the average execution timings using the following formula [40]:$t_{avg} = \frac{t_{10}+t_{20}+...+t_{100}}{10+20+...+100}$. We use the JUICE library for ABS signature generation and ABS signature verification [33]. Pairings are constructed on the curve $y^2 = x(x^2 + 1)$ over the field $F_p$ for some prime $p = 3(\text{mod } 4)$. Furthermore, the modular exponentiation algorithm and point multiplication operation are executed in multiplication cycle group with 1024-bit security parameter. The execution timings of individual cryptographic operations are presented in Table 2.

Table 3 summarizes the time cost of MBPA scheme in each phase according to the summarization in Table 2. Table 4 summarizes the computation cost required in other related schemes. Based on the implementation results in Tables 3 and 4, we analyze and compare the computation cost of related scheme, as shown in Fig4(a).

**TABLE 2.** **Computational notations.**

| Operation Times(ms) | Average Times | Description |
|---|---|---|
| $t_{bp}$ | 4.923ms | Execution time for one pairing operation in $G$ |
| $t_{sm}$ | 2.862ms | Execution time for scalar multiplication operation in $G$ |
| $t_{sg}$ | 44.016ms | Execution time for ABS signature generation operation |
| $t_{sv}$ | 33.86ms | Execution time for ABS signature verification operation |
| $t_{se}$ | 4.365ms | Execution time for symmetric encryption operation |
| $t_{sd}$ | 0.186ms | Execution time for symmetric decryption operation |
| $t_{hp}$ | 10.936ms | Execution time for hash-to-point operation |

**TABLE 3.** **Time cost during the design and implementation process of MBPA scheme.**

| Average Times(ms) cost of Phase | Time cost (ms) |
|---|---|
| Patient request phase | $4t_{sm}+t_{se}+t_{hp}\approx$26.749(ms) |
| Chain transaction phase (I) | (t+3)$t_{sm}$+2$t_{hp}\approx$ 2.862*t+30.485(ms) |
| Storage and response phase(I) | $2t_{sm}+t_{se}+t_{sd}\approx$10.275(ms) |
| Treatment phase | $6t_{sm}+t_{se}+t_{sd}+t_{hp}\approx$21.211(ms) |
| Chain transaction phase (II) | (t+3)$t_{sm}$+2$t_{hp}\approx$2.862*t+30.458(ms) |
| Storage and response phase(II) | $2t_{sm}+t_{se}+t_{sd}\approx$10.275(ms) |
| Checking up phase | $2t_{sm}+t_{sd}+t_{hp}\approx$16.846(ms) |

In patient request phase, Cheng et al's [12] protocol has to carry out three pairing operations, one scalar multiplication operation, one ABS signature verification operation, one symmetric encryption operation, one symmetric decryption operation and six hash-to-point operations. Therefore, the running time is $3t_{bp} + t_{sm} + t_{sv} + t_{se} + t_{sd} + 6t_{hp} \approx 121.658ms$. In treatment phase, it has to carry out two pairing operations, one scalar multiplication operation, one ABS signature generation operation, one ABS signature verification operation, one symmetric encryption operation, one symmetric decryption operation and five hash-to-point operations. Therefore, the running time is $2t_{bp} + t_{sm} + t_{sg} + t_{sv} + t_{se} + t_{sd} + 5t_{hp} \approx 147.785ms$. In checking up phase, it has to carry

**TABLE 4.** Average Times(ms) cost during the main phase of other related schemes.

| Average Times(ms) cost of Phase | [12] | [13] | [2] |
|---|---|---|---|
| Patient request phase | $3t_{bp}+t_{sm}+t_{sv}+t_{se}+t_{sd}+6t_{hp}\approx$ 121.658ms | $t_{sg}+t_{sv}+2t_{se}+t_{sd}+6t_{hp}\approx$ 152.408ms | $2t_{bp}+t_{sm}+t_{se}+t_{sd}+4t_{hp}\approx$ 63.865ms |
| Treatment phase | $2t_{bp}+t_{sm}+t_{sg}+t_{sv}+t_{se}+t_{sd}+5t_{hp}\approx$ 147.785ms | $t_{sg}+2t_{sv}+2t_{se}+2t_{sd}+6t_{hp}\approx$ 186.454ms | $2t_{bp}+t_{sm}+t_{se}+t_{sd}+4t_{hp}\approx$ 61.003ms |
| Checking up phase | $t_{bp}+t_{sm}+t_{sv}+t_{se}+t_{sd}+4t_{hp}\approx$ 80.94ms | $t_{sv}+t_{se}+t_{sd}+5t_{hp}\approx$ 93.091ms | $t_{bp}+t_{se}+t_{sd}+4t_{hp}\approx$ 53.218ms |

out one pairing operation, one scalar multiplication operation, one ABS signature verification operation, one symmetric encryption operation, one symmetric decryption operation and four hash-to-point operations. Therefore, the running time is $t_{bp} + t_{sm} + t_{sv} + t_{se} + t_{sd} + 4t_{hp} \approx 80.94ms$.

In patient request phase, Li et al's [13] protocol has to carry out one ABS signature generation operation, one ABS signature verification operation, one symmetric encryption operation, one symmetric decryption operation and six hash-to-point operations. Therefore, the running time is $t_{sg} + t_{sv} + 2t_{se} + t_{sd} + 6t_{hp} \approx 152.408\ ms$. In treatment phase, it has to carry out one ABS signature generation operation, two ABS signature verification operations, two symmetric encryption operations, two symmetric decryption operations and six hash-to-point operations. Therefore, the running time is $t_{sg} + 2t_{sv} + 2t_{se} + 2t_{sd} + 6t_{hp} \approx 186.454ms$. In checking up phase, it has to carry out one ABS signature verification operation, one symmetric encryption operation, one symmetric decryption operation and five hash-to-point operations. Therefore, the running time is $t_{sv}+t_{se}+t_{sd}+5t_{hp} \approx 93.091ms$.

In patient request phase, Liu et al's [2] protocol has to carry out two pairing operations, two scalar multiplication operations, one symmetric encryption operation, one symmetric decryption operation and four hash-to-point operations. Therefore, the running time is $2t_{bp} + 2t_{sm} + t_{se} + t_{sd} + 4t_{hp} \approx 63.865ms$. In treatment phase, it has to carry out two pairing operations, one scalar multiplication operation, one symmetric encryption operation, one symmetric decryption operation and four hash-to-point operations. Therefore, the running time is $2t_{bp}+t_{sm}+t_{se}+t_{sd}+4t_{hp} \approx 61.003ms$. In checking up phase, it has to carry out one pairing operation, one symmetric encryption operation, one symmetric decryption operation and four hash-to-point operations. Therefore, the running time is $t_{bp} + t_{se} + t_{sd} + 4t_{hp} \approx 53.218ms$.

In patient request phase, MBPA has to carry out four scalar multiplication operations, one symmetric encryption operation and one hash-to-point operation. Therefore, the running time is $4t_{sm}+t_{se}+t_{hp} \approx 26.749ms$. In chain transaction phase (I/II), it has to carry out $(t+3)$ scalar multiplication operations and two hash-to-point operations. Therefore, the running time is $(t + 3)t_{sm} + 2t_{hp} \approx 2.862 * t + 30.485ms$. In storage and response phase(I/II), it has to carry out two scalar multiplication operations, one symmetric encryption operation and one symmetric decryption operation. Therefore, the running time is $2t_{sm} + t_{se} + t_{sd} \approx 10.275ms$. In treatment phase, it has to carry out six scalar multiplication operations, one symmetric encryption operation, one symmetric decryption operation and one hash-to-point operation. Therefore, the running time is $6t_{sm} + t_{se} + t_{sd} + t_{hp} \approx 21.211ms$. In checking up phase, it has to carry out two scalar multiplication operations, one symmetric decryption operation and one hash-to-point operation. Therefore, the running time is $2t_{sm} + t_{sd} + t_{hp} \approx 16.846ms$.

**TABLE 5.** Communication cost among relevant authentication schemes.

| | [12] | [13] | [2] | MBPA |
|---|---|---|---|---|
| communication cost/bits | 12480 | 7520 | 9184 | 7360 |

According to the above comparisons of computation cost, it is clear that MBPA scheme has much less running time than other three related schemes [2], [12], [13] of these three algorithms.

### B. COMMUNICATION COST

In this subsection, we analyze and compare the communication costs of MBPA scheme and other three related schemes [2], [12], [13]. Without loss of generality, the size of the element in $G_p$ and $Z_q^*$ is 1024 bits and 160 bits, respectively. The length of random number is also 160 bits. The length of signature value is 1024 bits. The length of encryption value is 512 bits. The length of a patients identity is 32 bits. Sometimes, it is is dependent on its domain size. The length of the output of a hash function is dependent on its domain size. For a regular hash, we assume it to be 256 bits. The length of the timestamp is set as 32 bits. The comparisons among related schemes are listed in Table 5, as shown in Fig4(b).

In Cheng et al's [12] scheme, among the interactive messages, there are three elements in $G_p$, twelve elements in $Z_q^*$, three identities, four signature values, six encryption values and six timestamps. Therefore, the communication cost is $3^*1024 + 12^*160 + 4^*32 + 4^*1024 + 6^*512 + 6^*32 = 12480$ bits

In Li al's [13] scheme, among the interactive messages, there are six elements in $Z_q^*$, eight identities (160 bits), two signature values, six encryption values and one random number. Therefore, the communication cost is $6^*160 + 8^*160 + 2^*1024 + 6^*512 + 160 = 7520$ bits

In Liu's [2] scheme, among the interactive messages, there are two elements in $G_p$, three elements in $Z_q^*$, four identities, thirteen hash values, six encryption values and four timestamps. Therefore, the communication cost is $2^*1024 + 3^*160 + 4^*32 + 13^*512 + 6^*512 + 4^*32 = 9184$ bits

In MBPA scheme, among the interactive messages, there are four elements in $G_p$, four identities, four identities, four hash values, four encryption values and two timestamps. Therefore, the communication cost is $4^*1024 + 4^*32 + 4^*256 + 4^*512 + 4^*32 = 7360$ bits.

**TABLE 6. Security features comparison among relevant authentication schemes.**

| | [12] | [13] | [2] | MBPA |
|---|---|---|---|---|
| Completeness and mutual authentication | Yes | Yes | Yes | Yes |
| Patient anonymity | No | No | Yes | Yes |
| Patient traceability | No | Yes | Yes | Yes |
| Perfect forward secrecy | Yes | Yes | Yes | Yes |
| No verifier table | No | No | No | Yes |
| Birthday collision resilience | No | No | Yes | |
| Resilience to impersonation attack | Yes | Yes | Yes | Yes |
| Resilience to internal attacks | Yes | Yes | Yes | Yes |
| Resistance to replay attack | No | No | Yes | Yes |
| Resistance to man-in-the-middle attack | Yes | Yes | Yes | Yes |
| Provable security | No | Yes | Yes | Yes |
| secure channel in authentication phase | No | No | No | No |

According to the above comparisons of communication cost, we know that the MBPA also has much less communication cost than other three related schemes [2], [12], [13].

### C. SECURITY COMPARISONS

To show the security advantages of MBPA scheme, we present security comparisons between MBPA scheme and other related schemes [2], [12], [13]. The security comparisons are listed in Table 6. From Table 6, we can get that MBPA scheme can satisfy all ten security and function requirements. Therefore, the MBPA scheme is more secure than other three related schemes.

## VII. CONCLUSION AND ONGOING WORK

TMIS is not a far fetched concept, and in the complexity of future, TMIS will require a more robust security solution. In order to establish a secure remote user authentication based on blockchain in TMIS, we proposed a novel MediBchain-based framework (MBPA) in this paper. The MBPA scheme leverages the underpinning characteristics of MediBchain to realize a decentralized and privacy-preserving solution in TMIS. Specifically, we utilized double anonyms (one is used to anonymously authenticate patients, the other is used to trace malicious patients under necessary), shared value (to efficiently authenticate gateways in few encryption times), and certificateless cryptography (to provide confidentiality of the requested messages). We then demonstrated the security of MBPA and evaluated the performance of the prototype, theoretically. The results show that MBPA scheme is very suitable for computation-limited mobile devices compared with other related existing schemes. The future research is to fully identify the practical threats on MediBchain-based authentication schemes with better performance and evaluate the their performance using software and hardware.

## REFERENCES


[1] Y. Yang, X. Zheng, and C. Tang, "Lightweight distributed secure data management system for health Internet of Things," *J. Netw. Comput. Appl.*, vol. 89, pp. 26–37, Jul. 2017.

[2] X. Liu and W. Ma, "ETAP: Energy-efficient and traceable authentication protocol in mobile medical cloud architecture," *IEEE Access*, vol. 6, pp. 33513–33528, 2018.

[3] A. A. Alomar, M. Z. A. Bhuiyan, and A. Basu, S. Kiyomoto, and M. S. Rahman, "Privacy-friendly platform for healthcare data in cloud based on blockchain environment," *Future Gener. Comput. Syst.*, vol. 95, pp. 511–521, Jun. 2019.

[4] W. Jiang, H. Li, G. Xu, M. Wen, G. Dong, and X. Lin, "PTAS: Privacy-preserving thin-client authentication scheme in blockchain-based PKI," *Future Gener. Comput. Syst.*, vol. 96, pp. 185–195, Jul. 2019.

[5] S. Ahmad, S. Khan, and M. A. Kamal, "What is blockchain technology and its significance in the current healthcare system? A brief insight," *Current Pharmaceutical Des.*, vol. 25, pp. 496–503, Apr. 2019.

[6] C. Rolim, F. L. Koch, C. Westphall, J. Werner, A. Fracalossi, and G. S. Salvador, "A cloud computing solution for patient's data collection in health care institutions," in *Proc. 2nd Int. Conf. Ehealth, Telemed., Social Med.*, Feb. 2010, pp. 95–99.

[7] M. R. Patra, R. K. Das, and R. P. Padhy, "CRHIS: Cloud based rural healthcare information system," in *Proc. 6th Int. Conf. Theory Pract. Electron. Governance*, Oct. 2012, pp. 402–405.

[8] Y. Zhang, M. Qiu, C.-W. Tsai, M. M. Hassan, and A. Alamri, "Health-CPS: Healthcare cyber-physical system assisted by cloud and big data," *IEEE Syst. J.*, vol. 11, no. 1, pp. 88–95, Mar. 2017.

[9] C. Chen, T.-T. Yang, M.-L. Chiang, and T.-F. Shih, "A privacy authentication scheme based on cloud for medical environment," *J. Med. Syst.*, vol. 38, pp. 126–143, Nov. 2014.

[10] S. Chiou, Z. Ying, and J. Liu, "Improvement of a privacy authentication scheme based on cloud for medical environment," *J. Med. Syst.*, vol. 40, pp. 92–101, Apr. 2016.

[11] P. Mohit, R. Amin, A. Karati, G. P. Biswas, and M. K. Khan, "A standard mutual authentication protocol for cloud computing based health care system," *J. Med. Syst.*, vol. 41, pp. 41–50, Apr. 2017.

[12] Q. Cheng, X. Zhang, and J. Ma, "ICASME: An improved cloud-based authentication scheme for medical environment," *J. Med. Syst.*, vol. 41, pp. 41–44, Mar. 2017.

[13] C.-T. Li, D.-H. Shih, and C.-C. Wang, "Cloud-assisted mutual authentication and privacy preservation protocol for telecare medical information systems," *Comput. Methods Programs Biomed.*, vol. 157, pp. 191–203, Apr. 2018.

[14] L. Linn and M. Koo, "Blockchain for health data and its potential use in health it and healthcare related research," in *Proc. ONC/NIST Blockchain Healthcare Res.Workshop (ONC/NIST)*. Gaithersburg, MD, USA, 2016, pp. 254–267.

[15] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control," *J. Med. Syst.*, vol. 40, pp. 218–232, Oct. 2016.

[16] X. Xu, Q. Lu, Y. Liu, L. Zhu, H. Yao, and A. V. Vasilakos, "Designing blockchain-based applications a case study for imported product traceability," *Future Gener. Comput. Syst.*, vol. 92, pp. 399–406, Mar. 2019.

[17] M. Simic, G. Sladic, and B. Milosavljević, "A case study IoT and blockchain powered healthcare," in *Proc. 8th PSU-UNS Int. Conf. Eng. Technol. (ICET)*, Jun. 2017, pp. 1–4.

[18] A. Ekblaw, A. Azaria, J. Halamka, and A. Lippman, "A case study for blockchain in healthcare: 'MedRec' prototype for electronic health records and medical research data," in *Proc. IEEE Open Big Data Conf.*, vol. 13, Aug. 2016, pp. 13–24.

[19] Y. Chen, S. Ding, Z. Xu, H. Zheng, and S. Yang, "Blockchain-Based medical records secure storage and medical service framework," *J. Med. Syst.*, vol. 43, pp. 43–45, Jan. 2019.

[20] A. D. Dwivedi, G. Srivastava, S. Dhar, and R. Singh, "A decentralized privacy-preserving healthcare blockchain for IoT," *Sensors*, vol. 19, no. 2, pp. 326–338, Jan. 2019.

[21] S. Wang, S. Zhu, and Y. Zhang, "Blockchain-based mutual authentication security protocol for distributed RFID systems," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2018, pp. 74–77.

[22] L. Li, J. Liu, L. Cheng, S. Qiu, W. Wang, X. Zhang, and Z. Zhang, "Creditcoin: A privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2204–2220, Jul. 2018.

[23] M. Conti, M. Hassan, and C. Lal, "BlockAuth: BlockChain based distributed producer authentication in ICN," *Comput. Netw.*, vol. 164, Dec. 2019, Art. no. 106888.

[24] Y. Xu, J. Ren, G. Wang, C. Zhang, J. Yang, and Y. Zhang, "A blockchain-based nonrepudiation network computing service scheme for industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3632–3641, Jun. 2019.

[25] M. Kim, K. Park, S. J. Yu, J. Lee, Y. Park, S.-W. Lee, and B. Chung, "A secure charging system for electric vehicles based on blockchain," *Sensors*, vol. 19, no. 13, pp. 3028–3050, Jul. 2019.
[26] J. Wang, L. Wu, K.-K. R. Choo, and D. He, "Blockchain based anonymous authentication with key management for smart grid edge computing infrastructure," *IEEE Trans. Ind. Informat.*, to be published.
[27] H. Wang, D. He, Z. Liu, and R. Guo, "Blockchain-based anonymous reporting scheme with anonymous rewarding," *IEEE Trans. Eng. Manage.*, to be published.
[28] R. Guo, H. Shi, D. Zheng, C. Jing, C. Zhuang, and Z. Wang, "Flexible and efficient blockchain-based ABE scheme with multi-authority for medical on demand in telemedicine system," *IEEE Access*, vol. 7, pp. 88012–88025, 2019.
[29] F. Tang, S. Ma, Y. Xiang, and C. Lin, "An efficient authentication scheme for blockchain-based electronic health records," *IEEE Access*, vol. 7, pp. 41678–41689, 2019.
[30] 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 Gener. Comput. Syst.*, vol. 95, pp. 420–429, Jun. 2019.
[31] A. H. Mohsin, A. A. Zaidan, B. B. Zaidan, O. S. Albahri, A. S. Albahri, M. A. Alsalem, and K. I. Mohammed, "Based blockchain-PSO-AES techniques in finger vein biometrics: A novel verification secure framework for patient authentication," *Comput. Standards Interfaces*, vol. 66, Oct. 2019, Art. no. 103343.
[32] M. Castro and B. Liskov,"Practical byzantine fault tolerance," in *Proc. OSDI*. vol. 99, Feb. 1999, pp. 173–186.
[33] C. Lin, D. He, X. Huang, K.-K. R. Choo, and A. V. Vasilakos, "BSeIn: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0," *J. Netw. Comput. Appl.*, vol. 116, pp. 42–52, Aug. 2018.
[34] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2003, pp. 452–473.
[35] Y. Yang, X. Zheng, X. Liu, S. Zhong, and V. Chang, "Cross-domain dynamic anonymous authenticated group key management with symptom-matching for e-health social system," *Future Gener. Comput. Syst.*, vol. 84, pp. 160–176, Jul. 2018.
[36] M. Ma, D. He, H. Wang, N. Kumar, and K.-K. R. Choo, "An efficient and provably secure authenticated key agreement protocol for fog-based vehicular ad-hoc networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8065–8075, Oct. 2019.
[37] L. Wu, J. Wang, K. R. Choo, and D. He, "Secure key agreement and key protection for mobile device user authentication," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 2, pp. 319–330, Feb. 2019.
[38] X. Jia, D. He, N. Kumar, and K.-K. R. Choo, "A provably secure and efficient identity-based anonymous authentication scheme for mobile edge computing," *IEEE Syst. J.*, to be published.
[39] D. Huang, S. Misra, M. Verma, and G. Xue, "PACP: An efficient pseudonymous authentication-based conditional privacy protocol for VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 736–746, Sep. 2011.
[40] V. Odelu, S. Saha, R. Prasath, L. Sadineni, M. Conti, and M. Jo, "Efficient privacy preserving device authentication in WBANs for industrial e-health applications," *Comput. Secur.*, vol. 83, pp. 300–312, Jun. 2019.

**XIAOXUE LIU** received the M.S. degree from Shaanxi Normal University, Xi'an, China, in 2017. She is currently pursuing the Ph.D. degree in cryptography with the State Key Laboratory of Integrated Services Networks, School of Telecommunications Engineering, Xidian University, Xi'an. Her current research interests include cryptography and information security.

**WENPING MA** received the B.S. and M.S. degrees in fundamental mathematics from Shaanxi Normal University, Xi'an, China, in 1987 and 1990, respectively, and the Ph.D. degree in communication and information system from Xidian University, Xi'an, in 1999, where he is currently a Full Professor with the School of Telecommunications Engineering. His current research interests include information theory, communication theory, the error correcting code, and information security.

**HAO CAO** received the B.S. and M.S. degrees from Huaibei Normal University, in 2004 and 2007, respectively, and the Ph.D. degree from Xidian University, in 2019. Since 2018, he has been an Associate Professor with the School of Information and Network Engineering, Anhui Science and Technology University, Chuzhou, China. His research interests include quantum computation and quantum information, quantum cryptography, and information security.