





## Research Article

# Certificateless Public Auditing for Cloud-Based Medical Data in Healthcare Industry 4.0

Hui Tian <sup>1,2,3</sup> Weiping Ye <sup>1,2,3</sup> Jia Wang<sup>1,2,3</sup> Hanyu Quan <sup>1,2,3</sup>  
and Chin-Chen Chang <sup>4</sup>

<sup>1</sup>College of Computer Science and Technology, National Huaqiao University, Xiamen 361021, China

<sup>2</sup>Xiamen Key Laboratory of Data Security and Blockchain Technology, National Huaqiao University, Xiamen 361021, China

<sup>3</sup>Fujian Key Laboratory of Big Data Intelligence and Security, National Huaqiao University, Xiamen 361021, China

<sup>4</sup>Department of Information and Computer Science, Feng Chia University, Taichung 40724, Taiwan

Correspondence should be addressed to Hanyu Quan; [quanhanyu@hqu.edu.cn](mailto:quanhanyu@hqu.edu.cn)

Received 12 March 2023; Revised 20 September 2023; Accepted 6 November 2023; Published 27 November 2023

Academic Editor: Tao Li

Copyright © 2023 Hui Tian et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In the context of healthcare 4.0, cloud-based eHealth is a common paradigm, enabling stakeholders to access medical data and interact efficiently. However, it still faces some serious security issues that cannot be ignored. One of the major challenges is the assurance of the integrity of medical data remotely stored in the cloud. To solve this problem, we propose a novel certificateless public auditing for medical data in the cloud (CPAMD), which can achieve efficient batch auditing without complicated certificate management and key escrow. Specifically, in our CPAMD, a new secure certificateless signature method is designed to generate tamper-proof data block tags; a manageable delegated data outsourcing mechanism is presented to reduce the burden of data maintenance on patients and achieve auditability of outsourcing behavior; and a privacy-preserving augmented verification strategy is proposed to provide comprehensive auditing of both medical data and its source information without compromising privacy. We perform formal security analysis and comprehensive performance evaluation for CPAMD. The results demonstrate that the presented scheme can provide better auditing security and more comprehensive auditing capabilities while achieving good performance comparable to state-of-the-art ones.

## 1. Introduction

The vision of Industry 4.0 creates great potential for optimizing production and services in many industrial sectors, improving productivity, reliability, and flexibility [1]. This is especially true for healthcare, where cyber-physical systems, big data analytics, and cloud computing are revolutionizing the entire healthcare ecosystem and moving it toward Healthcare Industry 4.0 [2, 3]. The fundamental principle of Healthcare Industry 4.0 is to provide better services to patients and improve the effectiveness of the healthcare industry by connecting all medical components, including hospitals, medical professionals, and patients [1–3]. Today, Healthcare Industry 4.0 is profoundly transforming the healthcare field.

Owing to the continuously growing amounts of medical data under the background of healthcare industry 4.0, the traditional eHealth systems are overwhelmed. With powerful computing power and storage capacity, cloud technology has been widely introduced into the eHealth environment to manage patient data, maintain electronic health records (EHR), build knowledge bases, and monitor specific public health trends [4]. During the COVID-19 pandemic, the healthcare system extensively uses cloud computing to enable rapid deployment of applications in different organizations and efficient integration of data analysis among them [5]. Based on cloud computing, many organizational functions and clinical processes related to COVID-19, including monitoring, testing, triage, and diagnosis, have been efficiently implemented [5, 6]. In short,

cloud computing is playing an increasingly critical role in Healthcare Industry 4.0 as patients, medical staff, and hospitals all want to efficiently and securely obtain medical data and communicate within and across organizational boundaries [2, 5, 6].

Although there are many advantages of the cloud-based eHealth system, there are still some security issues in practical applications [7], which have raised widespread concerns from the government and society and spawned the promulgation of many related laws, such as Health Information Technology for Economic and Clinical Health Act (HITECH) and Health Insurance Portability and Accountability Act (HIPAA). One of the major challenges is to ensure the integrity of medical data remotely stored in the cloud [8–11], for which the reasons are twofold. First, cloud service provider (CSP) may hide the fact of data corruption for their own interests [8, 9]. Second, the user loses physical control over remotely stored data and cannot ensure data security through traditional means [10]. Once the medical data is corrupted, it would have a huge impact on medical treatment. Therefore, it is an important and critical task to ensure that medical cloud data remains intact and correct.

To check the integrity of data remotely stored, the cloud data auditing technique has come into being [8–11]. Generally, there are two implementation models, i.e., the private auditing model and the public auditing model. In the private auditing, the auditing process involves only the CSP and the user. It would increase the burden of user and produce a controversial auditing result. To address this issue, in the public auditing, an independent third-party auditor (TPA) is introduced to perform the auditing process, which greatly reduces the user's communication and computation burden and provides a reliable result. Therefore, public auditing has been widely adopted as the primary model in the latest auditing schemes [12–14].

For different application requirements and data types, a great many data auditing schemes have been proposed, including dynamic data auditing [8, 9, 15], privacy-preserving auditing [12], shared data auditing [13, 14], and multireplica auditing [16, 17]. More recently, identity-based (ID-based) signature has been introduced into the auditing schemes [18–20] to deal with the complex certificate management problem within public key infrastructure (PKI). Unfortunately, the ID-based signature has an inherent key escrow issue. Thus, certificateless signature, which requires no use of certificates and avoids the key escrow problem, has been widely introduced into data auditing schemes [21–26]. However, it is worth mentioning that none of the signature methods involved in the existing certificateless data auditing schemes [21–26] can provide strong enough security against public key replacement and master key attacks.

Although cloud data auditing has made significant progress, there are still some serious issues within medical cloud data that have not been well addressed.

The first is the inconsistency between the data owner (patient) and the data producer (medical personnel) in the medical cloud. Differing from common data generated and outsourced by the data owner, the medical data of patients

are generally produced by medical personnel. It is unrealistic for the patient to retrieve all medical data and then upload it to the cloud after processing. Therefore, it is crucial to design a delegated data outsourcing mechanism under patient management.

The second is the multisource nature of medical cloud data. Medical data of patients are generated by different medical professionals at various stages. If the traditional PKI-based auditing model is adopted, the management overhead of multiple proxies for certificate storage, distribution, and verification would impose a heavy burden on the eHealth system. In addition, the number of patients and healthcare professionals is too large to adopt identity-based cryptography. Thus, it is essential to develop a high-efficiency auditing protocol for multisource medical cloud data.

The third is the unique security requirements of medical data. On the one hand, in the event of a medical dispute, the auditable medical data and their source information (such as uploaders, data types, and uploading time) are the key to traceability and accountability. On the other hand, medical data are highly correlated with patient privacy. As a result, it is crucial to establish a comprehensive verification strategy with data privacy preservation.

To address the above issues, we present a novel Certificateless Public Auditing scheme for Medical cloud Data (called CPAMD), whose contributions can be summarized as follows:

- (1) We propose a public auditing scheme based on a novel secure certificateless signature method for medical cloud data, which provides necessary functions and satisfies security requirements for cloud-assisted eHealth systems.
- (2) To generate unforgeable data block tags, we propose a new secure certificateless signature method, which is proven to be secure against both type-I and type-II adversaries.
- (3) To solve the inconsistency problem between medical data owners and producers, we design a manageable delegated data outsourcing mechanism, which can not only significantly reduce computational and communicational burden on patients by authorizing medical personnel to outsource data but also support the verification of the outsourcing behavior.
- (4) Considering the multisource nature of medical cloud data, we develop an auditing algorithm based on a certificateless signature, which can achieve efficient batch auditing for patients' medical data handled by many medical personnel without complicated certificate management and key escrow.
- (5) To meet the unique security requirements of medical data, we establish an augmented data verification strategy with privacy protection, which designs an extended signature to provide comprehensive auditing for both medical data and their source information.

- (6) We analyze the security by proving the presented scheme can resist potential attacks and evaluate the performance by comparing it with the state-of-the-art ones theoretically and experimentally. The results demonstrate that CPAMD can provide better security and more comprehensive auditing functions while achieving desirable performance.

The structure of our work is listed as follows. Section 2 reviews some related work. Background and preliminaries are introduced in Section 3. Section 4 gives detailed description of CPAMD. Then, we perform the security analysis and fully performance evaluation in Sections 5 and 6, respectively. At last, Section 7 concludes this work.

## 2. Related Work

Healthcare Industry 4.0 takes advantage of cloud computing to store, process, and share massive amounts of medical data among patients, medical professionals, and various institutions. In spite of these benefits, it still faces serious security challenges. To address these issues, researchers have made arduous efforts. For example, Zhou et al. [27] came up with a secure dynamic medical data mining scheme with privacy preserving for cloud-assisted eHealth systems; Roy et al. [28] presented a fine-grained data access control protocol over multiple cloud servers in the healthcare industry 4.0. However, the integrity verification of medical cloud data, which is of great importance in cloud-based eHealth systems, has not been explored to a large extent.

In fact, for the traditional cloud data, cloud auditing has been generally adopted to verify and ensure data integrity. As a research foundation of this work, we would like to briefly review representative work on cloud data auditing. In 2007, Juels and Kaliski Jr [29] first presented proof of retrievability, a classical private auditing model involving only the CSP and user. Meanwhile, Ateniese et al. [30] proposed a provable data possession (PDP) scheme allowing a third party to perform the auditing process. In contrast, public auditing can alleviate the burden on the user and render a trustworthy auditing result, which is thereby considered more practical and reasonable.

With the continuous advancement of cloud services, a large number of auditing schemes for different data types (such as dynamic data, shared data, and multiple replicas) and specific application requirements (such as data privacy preserving) have emerged accordingly. The auditing schemes for dynamic data must not only check data integrity but also verify data freshness, for which various authenticated data structures are widely adopted, such as Merkle Hash Tree [8], Index-Hash Table [9], and Rank-based Authenticated Skip List [15]. To achieve privacy protection, it is often necessary to introduce a random masking technique to prevent the TPA from obtaining any data information while performing auditing [12]. For the shared data processed and maintained by various user groups, the auditing schemes should provide the data integrity checking as well as support group dynamics. To achieve this goal, various signature techniques that support multiuser operations, such as proxy

resignature [13] and ring signature [14], have been introduced successively. For multireplica data, the auditing scheme must ensure both the integrity of each copy and the correctness of the number of copies. The first multireplica auditing scheme is MR-PDP presented by Curtmola et al. [16]. Recently, some other advanced schemes have emerged, such as a dynamic multireplica auditing with different geographic locations [17].

Since the above cloud auditing schemes employed traditional public key cryptography (PKC), their key management relies heavily on the certificates generated by public key infrastructure (PKI). As the number of users continues to grow, the certificate management overhead becomes overwhelming. To overcome this problem, many identity-based data auditing schemes [18–20] have been presented, such as a privacy-preserving public auditing scheme using the zero-knowledge proof technique [18], a comprehensive data auditing scheme with delegated data outsourcing [19], and a light-weight data auditing scheme for health storage systems [20].

However, identity-based data auditing schemes face the challenge of key escrow. Particularly, the user-related block tags could be forged because the private key generator (PKG) gets the private key of every user. To overcome this challenge, Wang et al. [21] introduced certificateless signature into data auditing for the first time, where the user's private key includes two parts, namely the partial private key generated by the key generation center (KGC) and the secret value chosen by the user. Yang et al. [22] further proposed a certificateless-based PDP for shared data, which uses zero-knowledge and randomization to protect both data and user privacy. Li et al. [23] presented a certificateless public auditing scheme (CPIC) for group-shared data that can support user revocation effectively. Later, Gudeme et al. successively proposed two improved certificateless auditing schemes. One [24] aims to provide dynamic data auditing with data privacy protection, while the other [25] focuses on multireplica data auditing. More recently, Xu et al. [26] proposed a certificateless public auditing scheme (PAPD) for cloud-assisted medical wireless sensor networks (WSNs), enabling online sharing of medical data.

Unfortunately, the above-mentioned certificateless auditing schemes [21–26] still suffer from certain security risks, since the involved certificateless signature methods are not resistant to the public key replacement and master key attacks. To be more specific, a type-I adversary who replaces user's public key (owing to knowing the corresponding secret value) can derive user's partial key after making tag queries with the replaced public key, and a type-II adversary can generate the partial private key of the user with the master key. Based on the queried block-tag pair of  $i$ -th data block, it can easily convert it to another valid block-tag pair. Therefore, according to the actual needs of public auditing, it is of great necessity to develop a secure certificateless signature method that can withstand both type-I and type-II adversaries.

A thorough comparison of the presented scheme with several related schemes is shown in Table 1 in terms of certificate management, key escrow, delegated data

TABLE 1: Feature comparison with existing related work.

Features	PKI-based schemes			ID-based schemes			Certificateless-based schemes			
	[9]	[12]	[14]	[18]	[19]	[20]	[21]	[23]	[26]	CPAMD
Certificate management	×	×	×	√	√	√	√	√	√	√
Key escrow	√	√	√	×	×	×	√	√	√	√
Delegated data outsourcing	×	×	×	×	√	×	×	×	×	√
Comprehensive auditing	×	×	×	×	√	×	×	×	×	√

Note. For certificate management and key escrow, “√” means “no need” and “×” means “need”; for delegated data outsourcing and comprehensive auditing, “√” means “support” and “×” means “not support.”

outsourcing, and comprehensive auditing. In summary, cloud data auditing technology has made very significant progress. However, because of the peculiarities of medical cloud data, such as the inconsistency between data owners and producers, the multisource nature, and some unique security requirements, the previous schemes are not directly applicable to cloud-based eHealth system scenarios. Therefore, in this paper, we aim to propose a customized certificateless public auditing scheme to meet the necessary security requirements and provide comprehensive auditing functions for medical cloud data.

### 3. Background and Preliminaries

**3.1. System Model.** As illustrated in Figure 1, CPAMD has four types of entities, including cloud service provider (CSP), registry authority, user, and third-party auditor (TPA).

*Cloud Service Provider (CSP).* A semitrusted entity with enormous computing capabilities and storage capacities, provides medical data storage and management services for users.

*Registry Authority.* A trustworthy entity, which produces the partial private key for users, and in charge of setting up the system. Moreover, it stores the public parameters of outsourced medical record files. In practice, the health department can assume the role of the registry authority.

*User.* Includes patient and medical personnel. *Patient* is the data owner and *Medical personnel* is the data producer. Patient authorizes the designated medical personnel to outsource the medical data to CSP for storage and management, both of which can make an audit request to the TPA.

*Third Party Auditor (TPA).* An independent entity is authorized to check both data integrity and source information upon request.

Under normal circumstances, the CSP offers users with on-demand and reliable services. However, the self-interested CSP would hide the fact when data is damaged. In terms of the abuse and misuse of the delegation, malicious user may impersonate patient or authorized medical personnel to process and outsource medical data in undesirable ways. Furthermore, the TPA is deemed to be reliable but curious, which intends to obtain users’ data contents while conducting the auditing.

### 3.2. Security Assumptions

- (1) *Computational Diffie–Hellman (CDH) Assumption.* Let  $\mathbb{G}$  be a cyclic group of prime order  $p$ , for a randomly chosen generator  $g$  and random numbers  $a, b \in \mathbb{Z}_p^*$ , given  $(g, g^a, g^b) \in \mathbb{G}$ , it is computationally intractable to compute the value  $g^{ab}$ . That is, for any probabilistic polynomial-time adversary  $\mathcal{A}$ , the probability of solving the CDH problem is negligibly.
- (2) *Discrete Logarithm (DL) Assumption.* Let  $\mathbb{G}$  be a cyclic group of prime order  $p$  with a generator  $g$ , for a given  $h \in \mathbb{G}$ , it is computationally intractable to compute the value  $a \in \mathbb{Z}_p^*$ , such that  $h = g^a$ . In other words, for any probabilistic polynomial-time adversary  $\mathcal{A}$ , the probability of solving the DL problem is negligibly.

**3.3. Design Goals.** This work tries to realize the following goals to design a data auditing scheme for cloud-assisted eHealth systems with good security and efficiency:

- (1) *Public auditing.* Any third party authorized by the user can verify the data integrity and its source information.
- (2) *Designated authorization.* The delegation authorized by the patient can only be applied by the designated medical personnel. A new valid authorization cannot be forged.
- (3) *Multiproxy batch auditing.* The TPA can efficiently check the integrity of patients’ medical data that processed and outsourced by various different authorized medical workers simultaneously.
- (4) *Comprehensive auditing.* The TPA can verify both medical data integrity and its source information.
- (5) *Data privacy preserving.* The TPA learns nothing about patients’ data during the auditing process.

**3.4. Security Model.** Following the typical certificateless public auditing schemes [21–26], we consider three adversaries in this work, including type-I/II/III adversaries (namely,  $A_I$ ,  $A_{II}$ , and  $A_{III}$ ) with different attack capabilities. Both  $A_I$  and  $A_{II}$  intend to forge the block tag.  $A_{III}$  tries to pass the verification with forged auditing proof. The detailed definitions of  $A_I$ ,  $A_{II}$ , and  $A_{III}$  are shown as follows:

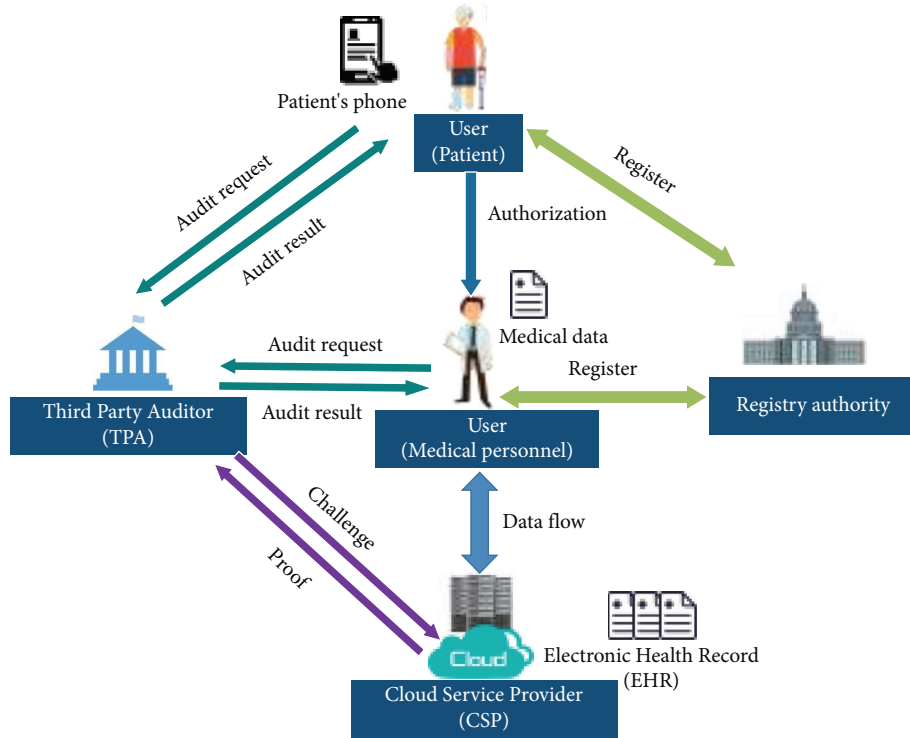


FIGURE 1: System model of the presented CPAMD.

- (1) Type-I adversary ( $A_I$ ):  $A_I$  can replace the user's public key, yet it cannot get the master key of registry authority.
- (2) Type-II adversary ( $A_{II}$ ):  $A_{II}$  can get the master key of registry authority, yet it cannot replace the public key.
- (3) Type-III adversary ( $A_{III}$ ):  $A_{III}$  tries to pass the verification with forged auditing proof.

The security of CPAMD is defined through three interactive games involving a challenger  $C$  and adversaries  $A_I$ ,  $A_{II}$ , and  $A_{III}$ .

*Game I.* This interactive game involves  $C$  and  $A_I$ .

*Setup.*  $C$  generates the master key  $s$  and public parameters  $Paras$ .  $C$  saves  $s$  in secret, transmits  $Paras$  to  $A_I$ .

*Queries.*  $A_I$  adaptively queries  $C$  in polynomial time.

- (1) Partial Private Key Query.  $A_I$  adaptively queries partial private key on  $ID$ .  $C$  generates partial key  $D_{ID}$ , transmits it to  $A_I$ .
- (2) SecretValue Query.  $A_I$  adaptively makes secret value query on  $ID$ .  $C$  generates the secret value  $S_{ID}$  and transmits it to  $A_I$ .
- (3) Public Key Query.  $A_I$  adaptively queries public key on  $ID$ .  $C$  generates the public key  $PK_{ID}$ , transmits it to  $A_I$ .
- (4) Public Key Replacement.  $A_I$  replaces the public key  $PK_{ID}$  of  $ID$  with  $PK_{ID}^*$  of its choice.

- (5) Tag-Query.  $A_I$  adaptively queries tag on  $(m, ID, PK_{ID})$ .  $C$  computes block tag  $\sigma$  and transmits it to  $A_I$ . *Forge.* Eventually,  $A_I$  generates a forged tag  $\sigma^*$  on  $(m^*, ID^*, PK_{ID}^*)$ .

$A_I$  is deemed to win, on the conditions that:

- (1) The forged tag  $\sigma^*$  is valid on  $(m^*, ID^*, PK_{ID}^*)$ .
- (2)  $A_I$  has not queried the partial key on  $ID^*$ .
- (3)  $A_I$  has not made the tag query on  $(m^*, ID^*, PK_{ID}^*)$ .

*Game II.* This interactive game involves the challenger  $C$  and adversary  $A_{II}$ .

*Setup.*  $C$  produces the master key  $s$  and public parameters  $Paras$ , transmits  $s$  and  $Paras$  to  $A_{II}$ .

*Queries.*  $A_{II}$  adaptively queries  $C$  in polynomial time.

- (1) SecretValue Query.  $A_{II}$  adaptively queries the secret value on  $ID$ .  $C$  calculates secret value  $S_{ID}$  and transmits it to  $A_{II}$ .
- (2) Public Key Query.  $A_{II}$  adaptively queries the public key on  $ID$ .  $C$  generates the public key  $PK_{ID}$ , transmits it to  $A_{II}$ .
- (3) Tag-Query.  $A_{II}$  adaptively queries tag on  $(m, ID, PK_{ID})$ .  $C$  computes block tag  $\sigma$  and transmits it to  $A_{II}$ . *Forge.* Eventually,  $A_{II}$  generates a forged tag  $\sigma^*$  on  $(m^*, ID^*, PK_{ID}^*)$ .

$A_{II}$  is deemed to win, on the conditions that:

- (1) The forged tag  $\sigma^*$  is valid on  $(m^*, ID^*, PK_{ID}^*)$ .
- (2)  $A_{II}$  has not made the secret value query on  $ID^*$ .
- (3)  $A_{II}$  has not made the tag query on  $(m^*, ID^*, PK_{ID}^*)$ .

*Definition 1.* If for any PPT adversary  $A_I$  and  $A_{II}$ , the probability of  $A_I$  and  $A_{II}$  winning Game I and II is negligible, the signature of the data block (i.e., block tag) is existentially unforgeable.

*Game III.* This interactive game involves  $C$  and  $A_{III}$ .

$A_{III}$  is regarded as a malicious CSP, which intends to deceive the auditor with a forged proof. Based on the Definition 1 above, no adversary can forge data block tags. Therefore, we focus on the issue that whether  $A_{III}$  is able to pass the verification using a forged auditing proof with incorrect data blocks.

*Setup.*  $C$  generates the master key  $s$  and public parameters  $Paras$ .  $C$  saves  $s$  in secret, transmits  $Paras$  to  $A_{III}$ .

*Tag-Query.*  $A_{III}$  adaptively queries for  $(m, ID, PK_{ID})$ .  $C$  computes block tag and transmits it to  $A_{III}$ .

*Challenge.*  $C$  generates a challenge  $chal$ , and then transmits it to  $A_{III}$ . Then,  $C$  requires  $A_{III}$  to respond to the  $chal$  with integrity proof  $P$ .

*Forge.*  $A_{III}$  generates a proof  $P$  in response to  $chal$ .

$A_{III}$  is deemed to win, if  $P$  passes the verification with the incorrect data blocks.

*Definition 2.* If for any PPT adversary  $A_{III}$ , the probability of  $A_{III}$  winning Game III is negligible, the auditing proof is existentially unforgeable.

**3.5. Cryptanalysis of Existing Certificateless Signature Schemes.** This subsection gives a detailed cryptanalysis of the certificateless signature schemes proposed in [21–26]. Since these certificateless signature schemes are basically the same, we take scheme [23] as an example to perform type-I and type-II attacks, respectively.

**3.5.1. Type-I Attack Analysis of Schemes [21–26].** As the above definition of type-I adversary,  $A_I$  can replace the user's public key  $PK_j$  with its chosen value, but it cannot get the system master key  $msk = s$ . According to the public key generation algorithm,  $PK_j = g^{x_j}$  is computed by the user  $ID_j$  using the secret value  $x_j$  that is secretly kept by the user. Therefore, when  $A_I$  performs the public key replacement attack, it can generate a secret value-public key pair  $(x'_j, PK'_j)$ , and then replace the user's public key  $PK_j$  with the new public key  $PK'_j$ , while  $A_I$  obtains the corresponding secret value  $x'_j$ . The detailed attack process is as follows:

- (1) A type-I adversary  $A_I$  adaptively generates a secret value-public key pair  $(x'_j, PK'_j)$ , and then replace the user's public key  $PK_j$  with the new public key  $PK'_j$ .
- (2)  $A_I$  queries the data block tag for  $(m_i, ID_j, PK'_j)$ . The challenger  $C$  generates the tag for the query by tag generation algorithm as

$$\sigma_i = D_j^{m_i} \cdot H_2(\omega_i)^{x'_j}. \quad (1)$$

and returns  $\sigma_i$  to  $A_I$ .

- (3) As  $A_I$  knows the secret value  $x'_j$  that corresponds to the replaced public key  $PK'_j$ , it can compute the value of  $H_2(\omega_i)^{x'_j}$ .
- (4) At last,  $A_I$  can compute the partial private key  $D_j$  of the user  $ID_j$  as follow:

$$\begin{aligned} D_j^{m_i} &= \frac{\sigma_i}{H_2(\omega_i)^{x'_j}} \\ \Rightarrow D_j &= \left( \frac{\sigma_i}{H_2(\omega_i)^{x'_j}} \right)^{1/m_i}. \end{aligned} \quad (2)$$

In conclusion, the certificateless signature scheme in the scheme [21–26] cannot resist the public key replacement attack, namely, a type-I adversary who replaces the user's public key can get to know the corresponding secret value since the public key is generated by the user from the secret value. Therefore, the adversary can extract the user's partial private key after making the tag query with the replaced public key.

**3.5.2. Type-II Attack Analysis of Schemes [21–26].** As the above definition of type-II adversary,  $A_{II}$  can get the master key  $msk = s$ , but it cannot replace the user's  $ID_j$  public key  $PK_j$ . According to the partial private key generation algorithm,  $D_j = H_1(ID_j)^s$  can be computed with the master key  $s$ . Then, based on the queried block-tag pair  $(m_i, \sigma_i)$  of  $i$ -th data block,  $A_{II}$  can convert it into another valid  $i$ -th block-tag pair  $(m'_i, \sigma'_i)$ . The detailed attack process is as follows:

- (1) Since a type-II adversary  $A_{II}$  knows the master key  $msk = s$ , it can obtain the partial private key  $D_j = H_1(ID_j)^s$  of the user  $ID_j$ .
- (2)  $A_{II}$  adaptively queries the tag  $\sigma_i$  for  $i$ -th data block  $m_i$ . The challenger  $C$  computes the tag for the query by tag generation algorithm as

$$\sigma_i = D_j^{m_i} \cdot H_2(\omega_i)^{x_j}. \quad (3)$$

and returns  $\sigma_i$  to  $A_{II}$ .

- (3)  $A_{II}$  has obtained the value of  $D_j^{m_i}$ , since it already gets the partial private key  $D_j = H_1(ID_j)^s$ . Then,  $A_{II}$  computes

$$H_2(\omega_i)^{x_j} = \frac{\sigma_i}{D_j^{m_i}}. \quad (4)$$

where  $\omega_i = \text{fid}||n||i$ , and  $\text{fid}$  denotes the unique file identity.

- (4) At last,  $A_{II}$  can convert the  $i$ -th block-tag pair into a valid forged block-tag pair  $(m'_i, \sigma'_i)$  as

$$\begin{aligned} \sigma'_i &= D_j^{m'_i} \cdot H_2(\omega_i)^{x_j} \\ &= D_j^{m'_i} \cdot \frac{\sigma_i}{D_j^{m_i}}. \end{aligned} \quad (5)$$

In conclusion, the certificateless signature in the previous scheme [21–26] cannot resist the master key attack, namely, a type-II adversary can generate the partial private key of the user based on the master key. For a block-tag pair  $(m_i, \sigma_i)$  of the  $i$ -th data block,  $A_{II}$  can easily convert it to another valid block-tag pair  $(m'_i, \sigma'_i)$ .

## 4. The Presented Scheme

**4.1. Overview of CPAMD.** Table 2 lists the primary notations in this work. In CPAMD, the registry authority produces the partial private key  $D_j$  for every user  $ID_j$  including patient  $ID_o$  and medical personnel  $ID_u$ . Meanwhile, every user selects a secret value  $x_j$  for himself/herself in private. The actual owner of medical data (i.e., patient)  $ID_o$  generates a valid authorization, which contains a pair of warrant and delegation  $(W_u, \delta_u)$ , for the designated medical worker  $ID_u$ . The authorization can prove the specific medical worker  $ID_u$  can represent the patient  $ID_o$  to process and upload the designated medical data within the prescribed time period.

The medical record file to be outsourced will be divided into multiple data blocks  $m_i$  ( $1 \leq i \leq n$ ), whose corresponding tags  $\sigma_i$  ( $1 \leq i \leq n$ ) are generated by the designated medical worker  $ID_u$ . It is worth noting that the warrant will be embedded into every block tag, which binds the designated medical worker's relevant information with the medical data.

Further, we leverage an extended signature [31] to establish an augmented verification strategy, where the patient signs the warrant as the medical worker's delegation and the delegated medical worker generate signatures of blocks as their metadata. In this way, the auditor can efficiently check both data integrity and their source information at the same time.

**4.2. A New Certificateless Signature Scheme.** To address the security issue of certificateless signatures in the previous schemes [21–26], we first present a new secure certificateless signature scheme, which includes the following six algorithms.

*Setup.* Registry authority (i.e., KGC) chooses two multiplicative cyclic groups  $\mathbb{G}_1$  and  $\mathbb{G}_2$  with the prime order  $p$ ,  $e: \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ , and secure hash functions  $H_1, H_2: \{0, 1\}^* \rightarrow \mathbb{G}_1$ ,  $H_3: \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$ , where  $g$  is a generator of  $\mathbb{G}_1$ . Then, it randomly chooses  $s \in \mathbb{Z}_p^*$  as the master key and produces system public key  $P_0 = g^s$ . Finally, the registry authority saves  $s$  privately and publishes  $Paras = (p, g, \mathbb{G}_1, \mathbb{G}_2, e, P_0, H_1, H_2, H_3)$ .

*Partial private key generation:* Upon receiving the identity  $ID_j$  of the user, registry authority computes  $D_j = H_1(ID_j)^s$  and returns it to the user.

*SecretValue generation:* The user  $ID_j$  chooses a random number  $x_j \in \mathbb{Z}_p^*$  as secret value.

*Public key generation:* The user  $ID_j$  utilizes secret value  $x_j$  to generate his/her public key  $PK_j = g^{x_j}$ .

*Tag generation:* The file  $F$  to be outsourced is separated into  $n$  data blocks. For each data block  $m_i \in \mathbb{Z}_p^*$  ( $1 \leq i \leq n$ ), the user  $ID_j$  chooses a random

number  $r_i \in \mathbb{Z}_p^*$ , and computes  $R_i = H_1(ID_u)^{r_i}$ . Then, the user  $ID_j$  generates the block tag as

$$\sigma_i = D_j^{r_i + m_i} \cdot H_2(\omega_i)^{x_j}, \quad (6)$$

where  $\omega_i = \text{fid} || n || m_i || i$ , and  $\text{fid}$  denotes the unique file identity. Finally, the user  $ID_j$  uploads the processed file  $\{F, \text{fid}, \{R_i\}_{1 \leq i \leq n}, \{\sigma_i\}_{1 \leq i \leq n}\}$  to the CSP.

*Tag verification:* The validation of block tag  $\sigma_i$  can be verified as

$$e(\sigma_i, g) = e(R_i \cdot H_1(ID_j)^{m_i}, P_0) \cdot e(H_2(\omega_i), PK_j). \quad (7)$$

The correctness of tag verification can be demonstrated as follows:

$$\begin{aligned} e(\sigma_i, g) &= e(D_j^{r_i + m_i} \cdot H_2(\omega_i)^{x_j}, g) \\ &= e(H_1(ID_j)^s \cdot (D_j^{r_i + m_i})^s, g) \cdot e(H_2(\omega_i)^{x_j}, g) \\ &= e(H_1(ID_j)^{r_i + m_i}, g^s) \cdot e(H_2(\omega_i), g^{x_j}) \\ &= e(R_i \cdot H_1(ID_j)^{m_i}, P_0) \cdot e(H_2(\omega_i), PK_j). \end{aligned} \quad (8)$$

**4.3. Detailed Construction of CPAMD.** The presented CPAMD consists of the following five processes, namely setup, registration, authorization, data outsourcing and auditing. The workflow of CPAMD is shown in Figure 2.

**4.3.1. Setup.** Registry authority sets up the system as described in the new certificateless signature method. Eventually, registry authority saves the master key  $s$  privately and publishes  $Paras = (p, g, \mathbb{G}_1, \mathbb{G}_2, e, P_0, H_1, H_2, H_3)$ .

**4.3.2. Registration.** Every user including patient and medical worker registers to registry authority to obtain the partial private key. This process consists of three algorithms as follows:

*Step 1 (Partial private key generation):* Upon receiving the identity  $ID_j$  of user (including patient  $ID_o$  and medical worker  $ID_u$ ), registry authority computes  $D_j = H_1(ID_j)^s$  and returns  $D_j$  to the user.

*Step 2 (SecretValue generation):* The user  $ID_j$  chooses a random number  $x_j$  as secret value.

*Step 3 (Public key generation):* The user  $ID_j$  utilizes secret value  $x_j$  to produce his/her public key  $PK_j = g^{x_j}$ .

**4.3.3. Authorization.** In order to authorize the designated medical worker  $ID_u$  to process and outsource medical data into cloud, the patient  $ID_o$  generates a warrant  $W_u = ID_o || ID_u || \text{DataType} || \text{TimeLimit} || \text{Institution}$ , where  $ID_o$  and  $ID_u$  are the identity of the patient and authorized medical worker,  $\text{DataType} \in \{0, 1\}^*$  is the specified type of medical data to be processed and outsourced,  $\text{TimeLimit} \in \{0, 1\}^*$  is the validity time period of the authorization, and

TABLE 2: Notations.

Notation	Description
$\mathbb{G}_1, \mathbb{G}_2$	Two multiplicative cyclic group
$H_1, H_2$	$H_1, H_2: \{0, 1\}^* \rightarrow \mathbb{G}_1$
$H_3$	$H_3: \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$
$s \in \mathbb{Z}_p^*$	The system master key
$P_0 \in \mathbb{G}_1$	The system public key
$n$	The number of data blocks
$m_i \in \mathbb{Z}_p^*$	The $i$ -th data block, $i \in [1, n]$
$\sigma_i \in \mathbb{G}_1$	The tag of data block $m_i$
$N_o$	The number of patients (i.e., data owner)
$N_u$	The number of medical workers (i.e., data producer)
$ID_o \in \{0, 1\}^*$	The identity of the $o$ -th patient, $o \in [1, N_o]$
$ID_u \in \{0, 1\}^*$	The identity of the $u$ -th medical worker, $u \in [N_o + 1, N_o + N_u]$
$D_j \in \mathbb{G}_1$	The partial private key of the $j$ -th user, $j \in [1, N_o + N_u]$
$x_j \in \mathbb{Z}_p^*$	The secret value of the $j$ -th user, $j \in [1, N_o + N_u]$
$PK_j \in \mathbb{G}_1$	The public key of the $j$ -th user, $j \in [1, N_o + N_u]$
$W_u \in \{0, 1\}^*$	The warrant for the $u$ -th medical worker, $u \in [N_o + 1, N_o + N_u]$
$\delta_u \in \mathbb{G}_1$	The delegation for the warrant $W_u$ , $u \in [N_o + 1, N_o + N_u]$

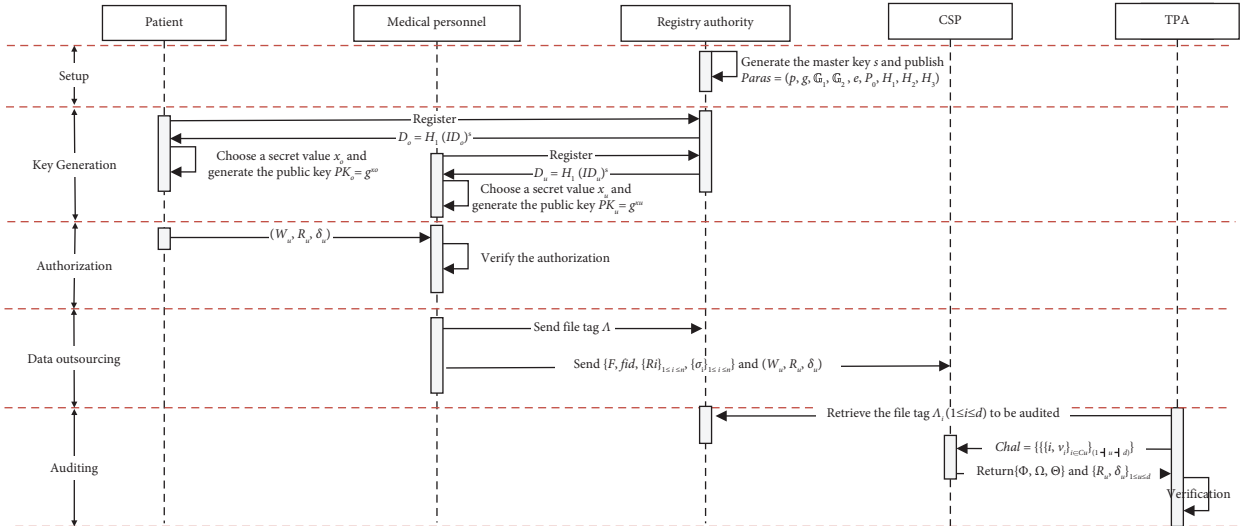


FIGURE 2: The workflow of CPAMD.

Institution  $\in \{0, 1\}^*$  is the medical location, where medical data are produced.

Furthermore, the patient  $ID_o$  randomly chooses  $r_u \in \mathbb{Z}_p^*$  and computes  $R_u = H_1(ID_o)^{r_u}$ . Then, the patient  $ID_o$  generates a signature on the warrant  $W_u$  as the delegation  $\delta_u$  for the designated medical worker  $ID_u$ :

$$\delta_u = D_o^{r_u + H_3(W_u)} \cdot H_2(\omega_u)^{x_o}, \quad (9)$$

where  $(D_o, x_o)$  is the private key of the patient,  $\omega_u = PK_o || PK_u$ . At last, the whole authorization  $(W_u, R_u, \delta_u)$  are transmitted to the specified medical worker  $ID_u$ . Upon receiving the authorization  $(W_u, R_u, \delta_u)$ , the medical worker  $ID_u$  would check the validity according to the following equation:

$$e(\delta_u, g) = e\left(R_u \cdot H_1(ID_o)^{H_3(W_u)}, P_0\right) \cdot e(H_2(\omega_u), PK_o). \quad (10)$$

If equation (10) holds, the medical worker  $ID_u$  accepts the authorization of the patient  $ID_o$ ; otherwise, the authorization fails.

**4.3.4. Data Outsourcing.** With the valid authorization  $(W_u, R_u, \delta_u)$ , the medical worker  $ID_u$  randomly selects  $fid \in \mathbb{Z}_p^*$  as the file identity of the medical file  $F$  to be outsourced, and divides  $F$  into  $n$  data blocks, namely,  $F = \{m_1, m_2, \dots, m_n\}$ . The file parameter is set as  $\lambda = fid || W_u || n$ . Then,  $ID_u$  generates the verification metadata as follows.



*Step 1 (File tag generation):* For the file  $F$  to be outsourced, the medical worker  $ID_u$  generates the file tag  $\Lambda = \lambda || S.Sig(\lambda, ssk) || spk$  with a one-time signature  $S = \langle Sig, Vrf \rangle$ , where  $(spk, ssk)$  is its public/secret key pair.

*Step 2 (Block tag generation):* For each data block  $m_i \in \mathbb{Z}_p^*$ , the medical worker  $ID_u$  selects a random number  $r_i \in \mathbb{Z}_p^*$ , computes  $R_i = H_1(ID_u)^{r_i}$  and generates the block tag as

$$\sigma_i = D_u^{r_i + m_i} H_2(\omega_i)^{x_u}, \quad (11)$$

where  $\omega_i = W_u || fid || n || m_i || i$ ,  $W_u$  is the warrant.

Finally, the authorized medical worker  $ID_u$  sends the file tag  $\Lambda$  to the registry authority, and transmits the processed medical file  $\{F, fid, \{R_i\}_{1 \leq i \leq n}, \{\sigma_i\}_{1 \leq i \leq n}\}$  together with its authorization  $(W_u, R_u, \delta_u)$  to the CSP.

After receiving these data, the CSP first checks the validation of the delegation  $\delta_j$  as equation (10). If it is invalid, the CSP rejects it; otherwise, the CSP verifies the validation of block tag  $\sigma_i$  as

$$e(\sigma_i, g) = e(R_i \cdot H_1(ID_u)^{m_i}, P_0) \cdot e(H_2(\omega_i), PK_u). \quad (12)$$

**4.3.5. Auditing.** CPAMD provides comprehensive auditing for both medical data and their source information. Both the patient and medical worker can initiate an auditing request, which can verify some specific medical files, but also check all medical records of the patient. Without loss of generality, we request the TPA to verify  $d$  medical files ( $fid_1, fid_2, \dots, fid_d$ ) processed by  $d$  different medical workers. The auditing process includes the following three steps:

*Step 1 (Challenge):* Upon request, the TPA first obtains the corresponding file tags  $\Lambda_i$  ( $1 \leq i \leq d$ ) from registry authority, then executes  $S.Vrf(\Lambda_i, spk)$  to verify it. If the verification fails, the file  $fid_i$  cannot be audited; otherwise, TPA executes following operations.

It first parses the file parameter  $\lambda = fid || W_u || n$  to get the warrant  $W_u$ , file identifier  $fid$  and block number  $n$ , which would be used to perform verification. Then, the TPA selects  $c$  data blocks from  $d$  medical files in random, to generate a challenge set  $I$ . For each  $i \in I$ , the TPA chooses a random number  $v_i \in \mathbb{Z}_p^*$  as its coefficient. To be specific, the set  $I$  consists of  $d$  subsets, that is,  $I = \{C_1, C_2, \dots, C_d\}$ , where the subset  $C_u$  is an index set of data blocks processed by the authorized medical worker  $ID_u$  ( $1 \leq u \leq d$ ). Eventually, the challenge  $chal = \{\{i, v_i\}_{i \in C_u}\}_{(1 \leq u \leq d)}$  are sent to the CSP.

*Step 2 (Proof generation):* In response to the challenge issued by the TPA, the CSP first computes the proof  $\Phi_u$ ,  $\Omega_u$ , and  $\Theta_u$  for each subset  $C_u$  as

$$\Phi_u = \prod_{i \in C_u} H_1(ID_u)^{m_i v_i}, \quad (13)$$

$$\Omega_u = \prod_{i \in C_u} \sigma_i^{v_i}, \quad (14)$$

$$\Theta_u = \prod_{i \in C_u} R_i^{v_i}. \quad (15)$$

Subsequently, the above proof  $\Phi_u$ ,  $\Omega_u$ , and  $\Theta_u$  related to each user  $ID_u$  are aggregated to get the final proof  $\Phi$ ,  $\Omega$ , and  $\Theta$  as

$$\Phi = \prod_{u=1}^d \Phi_u, \quad (16)$$

$$\Omega = \prod_{u=1}^d \Omega_u, \quad (17)$$

$$\Theta = \prod_{u=1}^d \Theta_u. \quad (18)$$

Eventually, the CSP transmits the proof  $P = \{\Phi, \Omega, \Theta\}$  and the corresponding delegations  $\{R_u, \delta_u\}_{1 \leq u \leq d}$  to the TPA.

*Step 3 (Verification):* After receiving a response from the CSP, the TPA first checks the validity of delegations  $\{R_u, \delta_u\}_{1 \leq u \leq d}$  by equation (10) to verify source information, which makes sure that the medical files were processed and outsourced by the designated medical workers as specified. The auditable source information can provide the credible evidence to help address medical disputes between the patient and medical worker.

Subsequently, the TPA checks the validity of the proof  $P = \{\Phi, \Omega, \Theta\}$  to verify data integrity as

$$e(\Omega, g) = e(\Theta \cdot \Phi, P_0) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right). \quad (19)$$

If equation (19) holds, the patient's medical data is perfectly stored; otherwise, the data are corrupted.

## 5. Security Analysis

### 5.1. Correctness

**Theorem 3.** *If the medical data stored in the CSP remain intact, the integrity proof can pass the verification.*

*Proof.* The correctness of the verification algorithm (equation (19)) is demonstrated as follows:

$$\begin{aligned}
e(\Omega, g) &= \prod_{u=1}^d e\left(\prod_{i \in C_u} \sigma_i^{v_i}, g\right) \\
&= \prod_{u=1}^d e\left(\prod_{i \in C_u} (D_u^{r_i+m_i} \cdot H_2(\omega_i)^{x_u})^{v_i}, g\right) \\
&= \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{s \cdot (r_i+m_i) \cdot v_i}, g\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{x_u \cdot v_i}, g\right) \\
&= \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{r_i v_i + m_i v_i}, g^s\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, g^{x_u}\right) \\
&= \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{r_i v_i} \cdot \prod_{i \in C_u} H_1(ID_u)^{m_i v_i}, g^s\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, g^{x_u}\right) \\
&= \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{r_i v_i}, g^s\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{m_i v_i}, g^s\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, g^{x_u}\right) \\
&= \prod_{u=1}^d e\left(\prod_{i \in C_u} R_i^{v_i}, P_0\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{m_i v_i}, P_0\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right) \\
&= e(\Theta, P_0) \cdot e(\Phi, P_0) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right) \\
&= e(\Theta \cdot \Phi, P_0) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right).
\end{aligned} \tag{20}$$

## 5.2. Soundness

**Theorem 4.** *The presented CPAMD is resistant to type-I adversary  $A_I$ , if the CDH assumption holds.*

*Proof.* Suppose that  $A_I$  wins Game I with a non-negligible probability  $\varepsilon$ , after performing the  $H_1$ -Query, *PartialPrivate Key-Query* and *Tag-Query* for  $q_{h1}$ ,  $q_{ppk}$  and  $q_t$  times, respectively, a simulator  $B$  can solve the CDH problem with probability  $Pr \geq (1-1/q_{h1})^{(q_{ppk}+q_t)} \cdot \varepsilon \cdot (1/q_{h1})$ . That is, with a problem instance  $(\mathbb{G}_1, g, g^a, g^b)$ ,  $B$  can compute  $g^{ab}$  with a non-negligible probability by running  $A_I$ .

*Setup.* The simulator  $B$  produces public parameters *Paras*, generates system public key as  $P_0 = g^a$ , where the master key  $s$  is equal to  $a$ .  $B$  sends *Paras* to  $A_I$ . Meanwhile, the master key is saved secretly,  $P_0 = g^a$  is available from the problem instance. At last,  $B$  randomly selects an identity  $ID^*$ .

*$H_1$ -Query.*  $B$  prepares a hash list  $L_1 = \{(ID, k, H_{ID})\}$ , in which  $L_1$  is empty initially.  $A_I$  adaptively makes hash queries for identity  $ID_i$ . When receiving the query,  $B$  checks whether the  $ID_i$  exist in  $L_1$ . If it does,  $B$  transmits  $H_{ID_i}$  to  $A_I$ ; else,  $B$  randomly selects a value  $k_i \in \mathbb{Z}_p^*$ , sets  $H_{ID_i}$  as

$$H_{ID_i} = \begin{cases} g^{k_i}, & ID_i \neq ID^*, \\ (g^b)^{k_i}, & ID_i = ID^*. \end{cases} \tag{21}$$

Then,  $B$  transmits  $H_{ID_i}$  to  $A_I$ , adds  $(ID_i, k_i, H_{ID_i})$  into  $L_1$ . Since  $k_i$  is a random value in  $\mathbb{Z}_p^*$ , both  $g$  and  $g^b$  are elements in  $\mathbb{G}_1$ , therefore,  $g^{k_i}$  and  $(g^b)^{k_i}$  have the identical distribution. As a result,  $A_I$  cannot tell the difference between results of  $H_{ID_i}$  returned by  $B$ .

*PartialPrivateKey-Query.*  $A_I$  adaptively makes partial key queries.  $B$  prepares a hash list  $L_2 = \{(ID, D_{ID}, PK_{ID}, S_{ID})\}$ , in which  $L_2$  is empty initially. For a partial private key query on  $ID_i$ ,  $B$  retrieves  $(ID_i, k_i, H_{ID_i})$  in the  $L_1$ . If  $ID_i = ID^*$ ,  $B$  aborts. Otherwise, according to the simulation,  $H_{ID_i} = g^{k_i}$ . The simulator  $B$  computes  $D_{ID_i} = (g^{k_i})^a$ . At last,  $B$  returns  $D_{ID_i}$  to  $A_I$  and adds  $(ID_i, D_{ID_i}, \perp, \perp)$  into  $L_2$ .

*SecretValue-Query.*  $A_I$  adaptively queries secret value for  $ID_i$ ,  $B$  checks whether  $S_{ID_i}$  exists in  $L_2$ . If it does,  $B$  returns  $S_{ID_i}$  to  $A_I$ ; else,  $B$  randomly chooses  $x_i \in \mathbb{Z}_p^*$  and sets  $S_{ID_i} = x_i$ ,  $PK_{ID_i} = g^{x_i}$ . At last,  $B$  transmits  $S_{ID_i}$  to  $A_I$ , adds  $(ID_i, D_{ID_i}, PK_{ID_i}, S_{ID_i})$  into  $L_2$ .

*PublicKey-Query.*  $A_I$  adaptively queries public key for  $ID_i$ ,  $B$  checks whether  $PK_{ID_i}$  exists in  $L_2$ . If it does,  $B$  returns  $PK_{ID_i}$  to  $A_I$ ; else,  $B$  randomly chooses  $x_i \in \mathbb{Z}_p^*$

and sets  $S_{ID_i} = x_i$ ,  $PK_{ID_i} = g^{x_i}$ . At last,  $B$  transmits  $PK_{ID_i}$  to  $A_I$ , adds  $(ID_i, D_{ID_i}, PK_{ID_i}, S_{ID_i})$  into  $L_2$ .

**PublicKey-Replacement.**  $A_I$  first selects a random number  $x_i^* \in \mathbb{Z}_p^*$ , sets  $S_{ID_i}^* = x_i^*$ ,  $PK_{ID_i}^* = g^{x_i^*}$ . Then,  $A_I$  adaptively executes the Public-Key-Replacement with the  $(ID_i, S_{ID_i}^*, PK_{ID_i}^*)$ . At last,  $B$  updates  $(ID_i, D_{ID_i}, PK_{ID_i}, S_{ID_i})$  to  $(ID_i, D_{ID_i}, PK_{ID_i}^*, S_{ID_i}^*)$  in  $L_2$ .

**$H_2$ -Query.**  $A_I$  adaptively makes the  $H_2$ -Query for information  $\omega \in \{0, 1\}^*$ .  $B$  prepares a hash list  $L_3 = \{(\omega, h, H_2(\omega))\}$ , in which  $L_3$  is empty initially. If  $\omega_i$  exists in  $L_3$ ,  $B$  transmits  $H_2(\omega_i)$  to  $A_I$ ; else,  $B$  randomly chooses  $h_i \in \mathbb{Z}_p^*$  and sets  $H_2(\omega_i) = g^{h_i}$ . Then  $B$  sends  $H_2(\omega_i)$  to  $A_I$ , adds  $(\omega_i, h_i, H_2(\omega_i))$  into  $L_3$ .

**Tag-Query.**  $A_I$  queries the tag for  $(ID_i, PK_{ID_i}, m_i, \omega_i)$ . If  $ID_i = ID^*$ ,  $B$  aborts. Otherwise,  $B$  retrieves  $D_{ID_i}$  and  $S_{ID_i}$  from  $L_2$ ,  $H_2(\omega_i)$  from  $L_3$ , and generates the tag  $\sigma_i$  for the query by tag generation algorithm. At last,  $B$  returns  $\sigma_i$  to  $A_I$ .

**Forge.** Eventually, the adversary  $A_I$  returns a forged tag  $\sigma^*$  on  $(m^*, ID^*, PK_{ID^*})$ .

**Analysis.** If  $A_I$  wins,  $B$  has  $ID_i = ID^*$ ,  $H_{ID^*} = (g^b)^{k^*}$ ,  $R^* = (g^b)^{k^* r^*}$  and  $H_2(\omega^*) = g^{h^*}$ . Then,  $B$  gets  $e(\sigma^*, g) = e(R^* \cdot H_1(ID^*)^{m^*}, P_0) \cdot e(H_2(\omega^*), PK_{ID^*})$  according to the verification algorithm. At last, we can compute the value of  $g^{ab}$  as

$$\begin{aligned}
e(\sigma^*, g) &= e(R^* \cdot H_1(ID^*)^{m^*}, P_0) \cdot e(H_2(\omega^*), PK_{ID^*}) \\
&= e(g^{bk^* r^*} \cdot g^{bk^* m^*}, g^a) \cdot e(g^{h^*}, PK_{ID^*}) \\
&= e(g^{bk^* (r^* + m^*)}, g^a) \cdot e(g^{h^*}, PK_{ID^*}) \\
&= e(g^{abk^* (r^* + m^*)}, g) \cdot e(PK_{ID^*}^{h^*}, g) \\
&= e(g^{abk^* (r^* + m^*)} \cdot PK_{ID^*}^{h^*}, g), \\
\Rightarrow \sigma^* &= g^{abk^* (r^* + m^*)} \cdot PK_{ID^*}^{h^*} \\
\Rightarrow g^{ab} &= \left( \frac{\sigma^*}{PK_{ID^*}^{h^*}} \right)^{1/k^* (r^* + m^*)}.
\end{aligned} \tag{22}$$

It means if the adversary  $A_I$  successfully generates a forged tag, then the CDH problem in  $\mathbb{G}_1$  (with  $g, g^a$  and  $g^b$ , compute  $g^{ab}$ ) can be solved by the simulator  $B$  that runs  $A_I$ .

Assume that the numbers of performing  $H_1$ -Query,  $Partial PrivateKey$ -Query,  $Tag$ -Query are  $q_{h1}$ ,  $q_{ppk}$  and  $q_t$ . Then, the probability of  $A_I$  winning Game I is evaluated as follows:

- (1)  $\zeta_1$ :  $B$  has not aborted Game I in the  $PartialPrivateKey$ -Query and  $Tag$ -Query.
- (2)  $\zeta_2$ :  $A_I$  successfully forged a valid tag  $\sigma$  on  $(m, ID, PK_{ID})$ .
- (3)  $\zeta_3$ : After  $\zeta_2$  occurs,  $ID_i$  is equal to  $ID^*$  (i.e.,  $ID_i = ID^*$ ).

From the simulation, we can obtain that

$$\Pr[\zeta_1] \geq \left(1 - \frac{1}{q_{h1}}\right)^{q_{ppk} + q_t},$$

$$\Pr[\zeta_2 | \zeta_1] \geq \varepsilon, \tag{23}$$

$$\Pr[\zeta_3 | \zeta_1 \wedge \zeta_2] \geq \frac{1}{q_{h1}}.$$

Therefore, the probability of simulator  $B$  solving the CDH problem is

$$\begin{aligned}
&\Pr[\zeta_1 \wedge \zeta_2 \wedge \zeta_3] \\
&= \Pr[\zeta_1] \cdot \Pr[\zeta_2 | \zeta_1] \cdot \Pr[\zeta_3 | \zeta_1 \wedge \zeta_2] \\
&\geq \left(1 - \frac{1}{q_{h1}}\right)^{q_{ppk} + q_t} \cdot \varepsilon \cdot \frac{1}{q_{h1}}.
\end{aligned} \tag{24}$$

□

**Theorem 5.** The presented CPAMD is resistant to type-II adversary  $A_{II}$ , if the CDH assumption holds.

**Proof.** Suppose that  $A_{II}$  wins Game II with a non-negligible probability  $\varepsilon$ , after performing  $SecretValue$ -Query and  $Tag$ -Query for  $q_s$  and  $q_t$  times, a simulator  $B$  can solve the CDH problem with  $\Pr \geq (1 - 1/q_s)^{(q_s + q_t)} \cdot \varepsilon \cdot (1/q_s)$ . That is, with a CDH problem instance  $(\mathbb{G}_1, g, g^a, g^b)$ ,  $B$  can compute the value of  $g^{ab}$  by running  $A_{II}$ .

**Setup.**  $B$  generates public parameters  $Paras$ , sets the master key as  $s$ .  $B$  sends  $s$  and  $Paras$  to  $A_{II}$ . Since  $A_{II}$  already has the master key,  $A_{II}$  has no need to query the partial private key. At last,  $B$  randomly selects an identity  $ID^*$ .

**$H_1$ -Query.**  $A_{II}$  adaptively makes  $H_1$ -Query for identity  $ID_i$ .  $B$  prepares a hash list  $L_1 = \{(ID, k, H_{ID})\}$ , in which  $L_1$  is empty initially. If the  $ID_i$  exists in  $L_1$ ,  $B$  returns  $H_{ID_i}$  to  $A_{II}$ ; else,  $B$  randomly chooses  $k_i \in \mathbb{Z}_p^*$ , computes  $H_{ID_i} = g^{k_i}$ . At last,  $B$  returns  $H_{ID_i}$  to  $A_{II}$  and adds  $(ID_i, k_i, H_{ID_i})$  to  $L_1$ .

**SecretValue-Query.**  $A_{II}$  adaptively makes secret value queries.  $B$  prepares a hash list  $L_2 = \{(ID, PK_{ID}, S_{ID})\}$ , in which  $L_2$  is empty initially. For a secret value query on  $ID_i$ ,  $B$  first retrieves  $(ID_i, PK_{ID_i}, S_{ID_i})$  in the  $L_2$ . If  $ID_i = ID^*$ ,  $B$  aborts. Otherwise,  $B$  randomly selects a number  $x_i \in \mathbb{Z}_p^*$  and sets  $S_{ID_i} = x_i$ ,  $PK_{ID_i} = g^{x_i}$ . Then  $B$  returns  $S_{ID_i}$  to  $A_{II}$ , adds  $(ID_i, PK_{ID_i}, S_{ID_i})$  to  $L_2$ .

**PublicKey-Query.**  $A_{II}$  adaptively makes  $PublicKey$ -Query for identity  $ID_i$ . When receiving the query,  $B$  checks whether  $(ID_i, PK_{ID_i}, S_{ID_i})$  exist in  $L_2$ . If it does,  $B$  transmits  $PK_{ID_i}$  to  $A_{II}$ ; else,  $B$  randomly chooses  $x_i \in \mathbb{Z}_p^*$ , sets  $PK_{ID_i}$  as

$$PK_{ID_i} = \begin{cases} g^{x_i}, & ID_i \neq ID^*, \\ (g^a)^{x_i}, & ID_i = ID^*. \end{cases} \tag{25}$$

$B$  transmits  $PK_{ID_i}$  to  $A_{II}$ , adds  $(ID_i, PK_{ID_i}, S_{ID_i})$  into  $L_2$ .

Since  $x_i$  is a random value in  $\mathbb{Z}_p^*$ ,  $g$  and  $g^a$  are elements in  $\mathbb{G}_1$ , therefore,  $g^{x_i}$  and  $(g^a)^{x_i}$  have the identical distribution. Thus,  $A_{II}$  cannot distinguish results of  $PK_{ID_i}$  returned by  $B$ .

*H<sub>2</sub>-Query.*  $A_{II}$  adaptively makes  $H_2$ -Query for information  $\omega_i \in \{0, 1\}^*$ .  $B$  prepares a list  $L_3 = \{(\omega, h, H_2(\omega))\}$ . If  $\omega_i$  exists in  $L_3$ ,  $B$  sends  $H_2(\omega_i)$  to  $A_{II}$ ; else,  $B$  randomly chooses  $h_i \in \mathbb{Z}_p^*$ , sets  $H_2(\omega_i) = (g^b)^{h_i}$ . Then,  $B$  transmits  $H_2(\omega_i)$  to  $A_{II}$ , adds  $(\omega_i, h_i, H_2(\omega_i))$  into  $L_3$ .

*Tag-Query.*  $A_{II}$  queries the tag for  $(ID_i, m_i, \omega_i)$ . If  $ID_i = ID^*$ ,  $B$  aborts; else  $B$  calculates  $D_{ID_i}$  by the master key, retrieves  $S_{ID_i}$  from  $L_2$  and  $H_2(\omega_i)$  from  $L_3$ , then generates the tag  $\sigma_i$  for  $(ID_i, m_i, \omega_i)$  by tag generation algorithm. At last,  $B$  returns  $\sigma_i$  to  $A_{II}$ .

*Forge.* Eventually, the adversary  $A_{II}$  returns a forged tag  $\sigma^*$  on  $(m^*, ID^*, PK_{ID^*})$ .

*Analysis.* If  $A_{II}$  wins,  $B$  has  $ID_i = ID^*$ ,  $PK_{ID^*} = (g^a)^{x^*}$ ,  $H_{ID^*} = g^{k^*}$ ,  $R^* = g^{k^*r^*}$  and  $H_2(\omega^*) = (g^b)^{h^*}$ .  $B$  gets  $e(\sigma, g) = e(R \cdot H_1(ID)^m, P_0) \cdot e(H_2(\omega), PK_{ID^*})$  according to the verification algorithm. At last, we can obtain

$$\begin{aligned}
e(\sigma^*, g) &= e(R^* \cdot H_1(ID^*)^{m^*}, P_0) \cdot e(H_2(\omega^*), PK_{ID^*}) \\
&= e(g^{k^*r^*} \cdot g^{k^*m^*}, g^s) \cdot e(g^{bh^*}, g^{ax^*}) \\
&= e(g^{k^*(r^*+m^*)}, g^s) \cdot e(g^{bh^*}, g^{ax^*}) \\
&= e(g^{k^*(r^*+m^*)s}, g) \cdot e(g^{abh^*x^*}, g) \\
&= e(g^{k^*(r^*+m^*)s} \cdot g^{abh^*x^*}, g), \\
\Rightarrow \sigma^* &= g^{k^*(r^*+m^*)s} \cdot g^{abh^*x^*} \\
\Rightarrow g^{ab} &= \left( \frac{\sigma^*}{g^{k^*(r^*+m^*)s}} \right)^{1/h^*x^*}.
\end{aligned} \tag{26}$$

It means if the adversary  $A_{II}$  successfully generates a forged tag, then the CDH problem in  $\mathbb{G}_1$  (with  $g, g^a$ , and  $g^b$ , compute  $g^{ab}$ ) can be solved by the simulator  $B$  that runs  $A_{II}$ .

Let  $q_s$  and  $q_t$  denote the numbers of *SecretValue-Query* and *Tag-Query*. The probability of  $A_{II}$  winning game is evaluated as follows:

- (1)  $\zeta_1$ :  $B$  has not aborted Game II in the *SecretValue-Query* and *Tag-Query*.

- (2)  $\zeta_2$ :  $A_{II}$  successfully generates a forged tag  $\sigma$  on  $(m, ID, PK_{ID})$ .
- (3)  $\zeta_3$ : After  $\zeta_2$  occurs,  $ID_i$  is equal to  $ID^*$  (i.e.,  $ID_i = ID^*$ ).

From which, we can obtain that

$$\Pr[\zeta_1] \geq \left(1 - \frac{1}{q_s}\right)^{q_s+q_t},$$

$$\Pr[\zeta_2 | \zeta_1] \geq \varepsilon, \tag{27}$$

$$\Pr[\zeta_3 | \zeta_1 \wedge \zeta_2] \geq \frac{1}{q_s}.$$

Therefore, the probability of the simulator  $B$  solving the CDH problem is

$$\begin{aligned}
&\Pr[\zeta_1 \wedge \zeta_2 \wedge \zeta_3] \\
&= \Pr[\zeta_1] \cdot \Pr[\zeta_2 | \zeta_1] \cdot \Pr[\zeta_3 | \zeta_1 \wedge \zeta_2] \\
&\geq \left(1 - \frac{1}{q_s}\right)^{q_s+q_t} \cdot \varepsilon \cdot \frac{1}{q_s}.
\end{aligned} \tag{28}$$

**Theorem 6.** *The presented CPAMD is resistant to type-III adversary  $A_{III}$ , if the DL assumption holds. That is, the auditing proof is existentially unforgeable.*

*Proof.* If  $A_{III}$  wins this game, a simulator  $B$  can break the DL problem. In other words, with a DL problem instance  $(\mathbb{G}_1, g, b = g^a)$ ,  $B$  can compute  $a \in \mathbb{Z}_p^*$  with a non-negligible probability by running  $A_{III}$ .

*Setup.*  $B$  generates the master key  $s$  and public parameters  $Paras$ .  $B$  saves  $s$  in secret, and sends  $Paras$  to  $A_{III}$ .

*Tag-Query.*  $A_{III}$  adaptively makes *Tag-Query* for  $(ID_i, PK_{ID_i}, m_i)$ .  $B$  generates the tag  $\sigma_i$  for the query by tag generation algorithm. At last,  $B$  returns  $\sigma_i$  to  $A_{III}$ .

*Challenge.* The simulator  $B$  sends a challenge  $chal = \{i, v_i\}_{i \in Cu} \cup \{1 \leq u \leq d\}$  to  $A_{III}$ .

*Forge.* In response to the challenge issued by  $B$ ,  $A_{III}$  should return a correct proof  $P = \{\Phi, \Omega, \Theta\}$ . However,  $A_{III}$  is deemed to win Game III, if  $A_{III}$  can pass the verification using a forged proof  $P^* = \{\Phi^*, \Omega^*, \Theta^*\}$ .

According to the verification algorithm, the auditing proof can be verified as

$$\begin{aligned}
e(\Omega, g) &= e(\Theta \cdot \Phi, P_0) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right) \\
&\Rightarrow \prod_{u=1}^d e\left(\prod_{i \in C_u} \sigma_i^{v_i}, g\right) = \prod_{u=1}^d e\left(\prod_{i \in C_u} R_i^{v_i}, P_0\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{m_i v_i}, P_0\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right),
\end{aligned} \tag{29}$$

$$\begin{aligned}
e(\Omega^*, g) &= e(\Theta^* \cdot \Phi^*, P_0) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right) \\
&\Rightarrow \prod_{u=1}^d e\left(\prod_{i \in C_u} \sigma_i^* v_i, g\right) = \prod_{u=1}^d e\left(\prod_{i \in C_u} R_i^* v_i, P_0\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_1(ID_u)^{m_i^* v_i}, P_0\right) \cdot \prod_{u=1}^d e\left(\prod_{i \in C_u} H_2(\omega_i)^{v_i}, PK_u\right).
\end{aligned} \tag{30}$$

Since  $A_{III}$  wins this game, we have  $\Omega = \Omega^*$  and  $\Theta = \Theta^*$ , but  $\Phi \neq \Phi^*$ . According to equations (29) and (30), and bilinear map's feature, we can obtain

$$\prod_{i \in C_u} H_1(ID_u)^{m_i v_i} = \prod_{i \in C_u} H_1(ID_u)^{m_i^* v_i}. \tag{31}$$

Set  $\Delta M_i = m_i v_i - m_i^* v_i$ , then, we can obtain

$$\prod_{i \in C_u} H_1(ID_u)^{\Delta M_i} = 1. \tag{32}$$

Given  $(\mathbb{G}_1, g, b = g^a)$ , we randomly select  $x_u, y_u \in \mathbb{Z}_p^*$ , and set  $H_1(ID_u) = g^{x_u} b^{y_u}$ . We can further obtain

$$\begin{aligned}
\prod_{i \in C_u} H_1(ID_u)^{\Delta M_i} &= \prod_{i \in C_u} (g^{x_u} b^{y_u})^{\Delta M_i} = 1 \\
&\Rightarrow g^{\sum_{i \in C_u} x_u \Delta M_i} b^{\sum_{i \in C_u} y_u \Delta M_i} = 1 \\
&\Rightarrow b = g^{-\left(\sum_{i \in C_u} x_u \Delta M_i / \sum_{i \in C_u} y_u \Delta M_i\right)} = g^a \\
&\Rightarrow a = -\frac{\sum_{i \in C_u} x_u \Delta M_i}{\sum_{i \in C_u} y_u \Delta M_i}.
\end{aligned} \tag{33}$$

It means if the adversary  $A_{III}$  successfully generates a forged auditing proof, then the DL problem in  $\mathbb{G}_1$  (i.e., given  $g, g^a$ , outputs  $a$ ) can be solved by the simulator  $B$  that runs  $A_{III}$ .

**Theorem 7** (Unforgeability of authorization). *If the CDH assumption holds, a valid new authorization is existentially unforgeable.*

*Proof.* The patient  $ID_o$  generates a signature on the warrant  $W_u$  as the delegation  $\delta_u$  for the designated medical worker  $ID_u$ :

$$\delta_u = D_o^{r_u + H_3(W_u)} \cdot H_2(\omega_u)^{x_o}. \tag{34}$$

The complete authorization  $(W_u, R_u, \delta_u)$  is sent to the specified medical worker  $ID_u$ . According to Definition 1, the signature in our scheme is existentially unforgeable if the CDH assumption holds. Thus, a new valid authorization cannot be forged.  $\square$

### 5.3. Privacy

**Theorem 8** (Data privacy preserving). *In CPAMD, the TPA is unable to learn any data information while performing the auditing process, if the DL assumption holds.*

*Proof.* While performing the auditing process, the TPA intends to learn the user's data information from  $P = \{\Phi, \Omega, \Theta\}$  returned by the CSP. Since  $\Theta$  is the aggregated value of random number  $r_i$ , which does not contain any data content, so we only need to show that both  $\Phi$  and  $\Omega$  in the proof do not leak any data information of the user.

First, according to equations (13) and (16) in the proof generation, we can obtain

$$\begin{aligned}
\Phi &= \prod_{u=1}^d \Phi_u \\
&= \prod_{u=1}^d \prod_{i \in C_u} H_1(ID_u)^{m_i v_i} \\
&= \prod_{u=1}^d H_1(ID_u)^{\sum_{i \in C_u} m_i v_i}.
\end{aligned} \tag{35}$$

According the DL problem in  $\mathbb{G}_1$ , it is infeasible to extract the information of  $\sum m_i v_i$ . Therefore, the TPA is unable to learn any data information from  $\Phi$ .

Second, according to equations (14) and (17), we can obtain

$$\begin{aligned}
\Omega &= \prod_{u=1}^d \Omega_u \\
&= \prod_{u=1}^d \prod_{i \in C_u} \sigma_i^{v_i} \\
&= \prod_{u=1}^d \prod_{i \in C_u} (D_u^{r_i + m_i} \cdot H_2(\omega_i)^{x_u})^{v_i} \\
&= \prod_{u=1}^d (D_u^{\sum_{i \in C_u} r_i v_i + \sum_{i \in C_u} m_i v_i}) \prod_{i \in C_u} H_2(\omega_i)^{x_u v_i}.
\end{aligned} \tag{36}$$

From the equation above,  $r_i$  is chosen randomly by the user, and the partial private key  $D_u$  is kept secretly; thereby, both  $r_i$  and  $D_u$  are unknown to the TPA. What is more, according to the DL problem in  $\mathbb{G}_1$ , the TPA cannot extract the information of  $\sum m_i v_i$ . Thus, the CPAMD can protect data privacy from the TPA.

As proven above, the presented CPAMD can satisfy all the security requirements. Moreover, we compare the security of our CPAMD with existing certificateless data auditing schemes [21–26]. The results are illustrated in Table 3. The presented CPAMD is resistant to both the public key replacement attack and master key attacks. Moreover, the TPA learns nothing about patients' data while performing the auditing process.

## 6. Performance Evaluation

In this section, we conduct theoretical analysis in terms of auditing functions, communication and computation costs, and then evaluate the performance through detailed comparative experiments with state-of-the-art schemes.

### 6.1. Theoretical Analysis

**6.1.1. Function comparison.** We first compare the presented CPAMD with state-of-the-art schemes (that is, IBDO [19], CPIC [23], and PAPD [26]) in terms of auditing functions. As shown in Table 4, our CPAMD is the only scheme that provides all functions: public auditing, secure certificateless signature, delegated data outsourcing, efficient batch auditing, and data privacy preserving.

In cloud-assisted eHealth systems, the patient's medical data are generated by different medical workers at various stages, so the delegated data outsourcing and efficient batch auditing are indispensable functions to achieve high efficiency. In addition, for security requirement, a secure signature method and data privacy preserving are necessary.

**6.1.2. Communication Costs.** Table 5 summarizes communication costs in the auditing. In the challenge phase, both CPAMD and PAPD support the multiproxy batch auditing, in which the challenge set is composed of  $d$  subsets, each of which is an index set of data blocks processed by the authorized medical worker  $ID_u$  ( $1 \leq u \leq d$ ). Therefore, the communication costs of both CPAMD and PAPD are  $c \cdot (|\mathbb{Z}_p^*| + |N|)$ . By contrast, CPIC only sends the count of challenged block and two random numbers ( $2|\mathbb{Z}_p^*| + |N|$ ) to launch a challenge. However, the CSP and TPA need to use pseudorandom permutation and pseudorandom function to compute real challenge set. That is, CPIC reduces communication overhead at the cost of computation overhead.

In the response phase, CPAMD's communication costs are only  $3|\mathbb{G}_1|$ , which are much lower than IBDO, CPIC, and PAPD. It should be noted that we conduct the comparison of communication costs under the multiproxy environment. Since IBDO does not mention specific operations with multiple proxies, we expand it into a multiproxy auditing scheme by generating the corresponding proof for each

medical worker. Therefore, the communication costs of IBDO during the response are linear to the number of authorized medical workers. In addition, PAPD introduces a blinding factor for each subset to protect data privacy, whose communication cost is  $2d \cdot |\mathbb{G}_1| + d \cdot |\mathbb{Z}_p^*|$ . By comparison, the presented CPAMD aggregates the data proof and tag proof from all involved medical workers to achieve batch auditing, thereby making the communication costs independent of the number of authorized medical workers.

**6.1.3. Computation Costs.** The computation costs of four schemes are given in Table 6. The tag generation costs of CPIC, PAPD, and CPAMD are similar and lower than those of IBDO. To be specific, CPIC, PAPD, and CPAMD all employ a certificateless signature to generate data block tags, among which the costs of CPIC and PAPD are the same, and CPAMD is slightly higher than these two schemes. However, CPIC and PAPD are not resistant to public key replacement and master key attacks. To address this issue, CPAMD introduces a nonce to protect the partial key, which requires an additional exponentiation on  $\mathbb{G}_1$ . Therefore, the tag generation costs of CPAMD are  $n \cdot d \cdot (H + 3E_1 + M_1)$ .

The proof generation costs of CPAMD are  $3c \cdot E_1 + 3(c - 1) \cdot M_1$ , for which the reasons are threefold. First, CPAMD generates an additional proof related to the nonce to resist public key replacement attacks. Second, it produces the data proof as  $H_1(ID_u)^{m_i v_i}$ , instead of  $m_i v_i$ , which can protect data privacy from the TPA. Third, to support efficient batch auditing, it aggregates the proof from all authorized medical workers. Overall, the proof generation costs of CPAMD are no greater than those of IBDO for the multiuser configuration (i.e.,  $d \geq 3$ ), but larger than those of CPIC and PAPD. However, we believe it is worthwhile to appropriately add some computational overhead in the proof generation to the CSP, because CPAMD can provide better security and comprehensive auditing functions, particularly secure certificateless signature, data privacy preserving and multiproxy efficient batch auditing.

In addition, the verification costs of CPAMD are  $(c - d + 1) \cdot M_1 + (d + 2) \cdot P + c \cdot E_1 + c \cdot H + (d - 1) \cdot M_2$ , which is the lowest among four schemes. As a result, the TPA can efficiently verify the patient's medical data that handled by multiple authorized medical workers simultaneously.

**6.2. Comparative Experiments.** We evaluate the performance through detailed comparative experiments on a Dell workstation with an Intel Xeon E3-1225 CPU at 3.31 GHz, 8 GB of RAM, and 7200 RPM Serial ATA drive, as well as a Linux system. Meanwhile, all algorithms are implemented based on the Charm-Crypto Library v0.43 [32]. We employ an MNT d159 curve, which has a 160 bit group order. All the statistical results are the averages of 20 trials.

To comprehensively evaluate the performance in the tag generation phase, two kinds of experimental setting have been established, in one of which, data block number is set as 5000 and data block size ranges from 1 KB to 128 KB. In the other one, the size of the data block is set as 4 KB, and the data block number increases from 5000 to 50000 at 5000 intervals.

TABLE 3: Security comparison with existing certificateless data auditing schemes.

Schemes	Correctness	Public key replacement attack	Master key attack	Data privacy preserving
Wang et al. [21]	√	×	×	×
Yang et al. [22]	√	×	×	√
CPIC [23]	√	×	×	×
Gudeme et al. [24]	√	×	×	√
Gudeme et al. [25]	√	×	×	√
PAPD [26]	√	×	×	√
CPAMD	√	√	√	√

Note. “√” means “support”; “×” means “not support.”

TABLE 4: The comparison of data auditing functions.

Schemes	Public auditing	Secure certificateless signature	Delegated data outsourcing	Efficient batch auditing	Data privacy preserving
IBDO [19]	√	×	√	—	×
CPIC [23]	√	×	×	√	×
PAPD [26]	√	×	×	√	√
CPAMD	√	√	√	√	√

Note. “√” denotes “support”; “×” denotes “not support”; “—” denotes “not mentioned.”

TABLE 5: The comparison of communication costs.

Scheme	Challenge	Response
IBDO [19]	$c \cdot ( \mathbb{Z}_p^*  +  N )$	$d \cdot (c \cdot  \mathbb{Z}_p^*  + 2 \mathbb{G}_1 )$
CPIC [23]	$2 \mathbb{Z}_p^*  +  N $	$d \cdot  \mathbb{G}_1  + d \cdot  \mathbb{Z}_p^* $
PAPD [26]	$c \cdot ( \mathbb{Z}_p^*  +  N )$	$2d \cdot  \mathbb{G}_1  + d \cdot  \mathbb{Z}_p^* $
CPAMD	$c \cdot ( \mathbb{Z}_p^*  +  N )$	$3 \mathbb{G}_1 $

Note.  $c$  is challenged block number;  $d$  is authorized medical personnel number;  $|N|$  is the element size in  $[1, n]$ ;  $|\mathbb{Z}_p^*|$  is the element size in  $\mathbb{Z}_p^*$ ;  $|\mathbb{G}_1|$  is the element size in group  $\mathbb{G}_1$ .

Figures 3 and 4, respectively, show the experimental results for these two settings, from which we can observe: (1) the tag generation time of the four schemes is all linear with the block size and block number; (2) the tag generation time of CPIC and PAPD are same; (3) for the same block size and block number, CPAMD is slightly more computationally expensive than CPIC and PAPD, all three of which are much lower than IBDO; and (4) with a fixed number of data blocks (i.e., 5000), the difference in tag generation overhead between CPAMD and CPIC is constant (about 20s), which does not increase with the block size. The reason for this phenomenon is that, compared with CPIC and PAPD, CPAMD employs a more secure certificateless signature method to resist the public key replacement and master key attacks. For each data block, CPAMD requires an additional exponential operation to compute an auxiliary verification parameter that is independent of the block size. For better security performance, we believe that a little extra computational overhead is worth it and necessary.

In addition, CPAMD provides a delegated data outsourcing mechanism, through which the patient can delegate designated medical workers to process and outsource the medical data. That is to say, in the presented CPAMD, the tag generation operations are not performed by patients but by the authorized medical personnel with professional equipment.

Therefore, we could safely make the following conclusions: first, compared with IBDO, which also supports the delegated data outsourcing, the performance of CPAMD is better than that of IBDO in the tag generation; second, compared to CPIC and PAPD, CPAMD trades a bit of computational overhead for better security and the support for delegated data outsourcing.

Figure 5 depicts the verification time of the TPA for different numbers of authorized medical personnel. In this experiment, the block size and block number are, respectively, set to 4 KB and 5000. In addition, the challenged blocks number is fixed at 460, and the number of authorized medical personnel increases from 10 to 100 with the interval of 10. The experimental results in Figure 5 demonstrate that: (1) the verification time of four schemes is all linear to the number of authorized medical personnel and (2) the verification time of CPAMD is lower than that of IBDO, CPIC, and PAPD. It is worth noting that IBDO, which supports the delegated data outsourcing, does not mention verification in a multiproxy environment. To perform this comparative experiment, we extend IBDO to implement a multiproxy auditing by executing verification for outsourced data from different authorized medical workers separately. CPIC supports efficient public auditing for data shared in a group, which provides batch auditing for multiple group users. However, a curious TPA or an external attacker can deduce the content of outsourced data by collecting linear combinations of challenged blocks in auditing process. To address this issue, PAPD introduces a blinding factor to protect data privacy. As a result, the verification time of PAPD is slightly higher than that of CPIC.

By contrast, CPAMD issues a challenge for all authorized medical workers. The CSP first computes the integrity proof for each proxy, and then aggregate them to get the final proof. In addition, the TPA cannot obtain any data information from the proof, thus protecting patient’s data

TABLE 6: The comparison of computation costs.

Scheme	Tag generation	Proof generation	Verification
IBDO [19]	$n \cdot d \cdot (H + (l + 3) \cdot E_1 + (l + 2) \cdot M_1)$	$d \cdot c \cdot E_1 + d \cdot (c - 1) \cdot M_1$	$d \cdot (4P + (c + l) \cdot M_1 + (l + c + 1) \cdot E_1 + 2E_2 + c \cdot H)$
CPIC [23]	$n \cdot d \cdot (H + 2E_1 + M_1)$	$c \cdot E_1 + (c - d) \cdot M_1$	$(c + d - 2) \cdot M_1 + (d + 2) \cdot P + (c + d) \cdot E_1 + (c + d) \cdot H + (d - 1) \cdot M_2$
PAPD [26]	$n \cdot d \cdot (H + 2E_1 + M_1)$	$(c + d) \cdot E_1 + (c - d) \cdot M_1$	$(c + 2d - 2) \cdot M_1 + (d + 2) \cdot P + (c + d) \cdot E_1 + (c + d) \cdot H + (d - 1) \cdot M_2$
CPAMD	$n \cdot d \cdot (H + 3E_1 + M_1)$	$3c \cdot E_1 + 3(c - 1) \cdot M_1$	$(c - d + 1) \cdot M_1 + (d + 2) \cdot P + c \cdot E_1 + c \cdot H + (d - 1) \cdot M_2$

Note.  $c$  represents the challenged block number;  $d$  represents user number;  $n$  represents data block number;  $H$  represents the operation time of hash function;  $E_1$  and  $E_2$  represent the exponential operation time on groups  $G_1$  and  $G_2$ ;  $M_1$  and  $M_2$  represent the multiplication operation time on groups  $G_1$  and  $G_2$ ;  $P$  represents the pairing operation time;  $l$  represents the length of security factor.

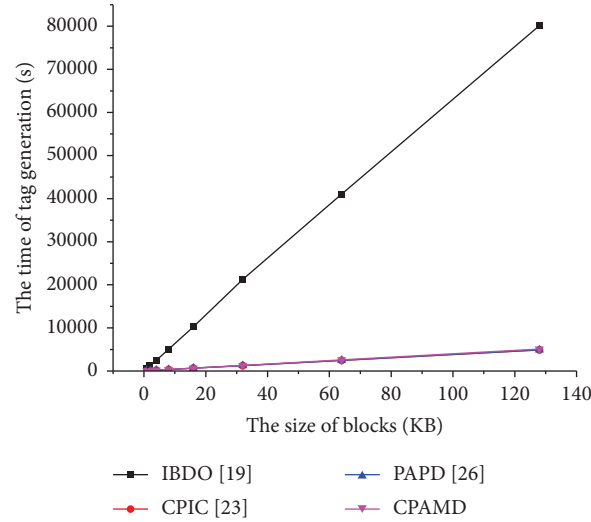


FIGURE 3: The time of tag generation for blocks in different sizes.

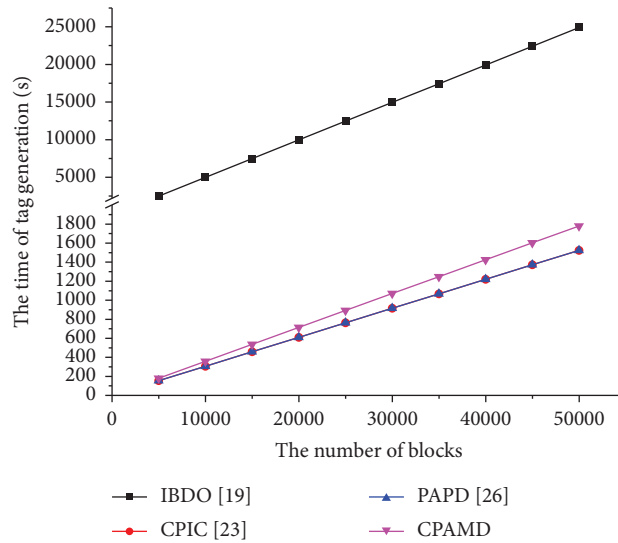


FIGURE 4: The time of tag generation for blocks in different numbers.



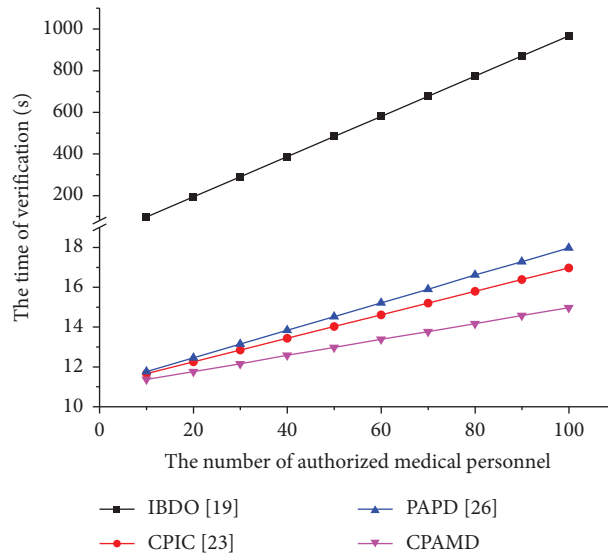


FIGURE 5: The time of verification for authorized medical personnel in different numbers.

privacy. To sum up, in the verification process, CPAMD outperforms IBDO, CPIC, and PAPD in terms of efficiency and security.

## 7. Conclusion

Aiming at the unique attributes and security requirements of cloud-based medical data in healthcare industry 4.0, a new certificateless public auditing scheme is presented. To generate unforgeable data block tags, a new secure certificateless signature method is developed, which is proven to be secure against type-I and type-II adversaries. To solve the inconsistency between medical data owners and producers, a manageable delegated data outsourcing mechanism is designed, which can relieve the burden on patients and verify the outsourcing behaviors of medical workers. Taking the multisource nature of medical cloud data into account, an auditing protocol based on certificateless signature is proposed to achieve efficient batch verification for medical data handled by various medical workers without complicated certificate management and key escrow. In addition, an augmented data verification strategy with privacy protection is presented to achieve comprehensive auditing for both medical data and their source information without leaking data privacy. The security analysis and performance evaluation results show that CPAMD can provide better auditing security and more comprehensive auditing functions while delivering good data auditing performance comparable to the state-of-the-art ones.

With the increasingly rich applications of eHealth systems, the auditing research on medical cloud data will be further deepened. For example, with the increasing popularity of collaborative medical care, patients' EHRs are shared among authorized medical personnel, which places new requirements on data auditing, such as efficient dynamic group management and identity traceability. It is of great significance to provide a tailor-made solution for

public auditing of shared medical data in the cloud, which is one of the directions of our future efforts.

## Data Availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFC3304501, Natural Science Foundation of Fujian Province of China under Grant 2020J05059, the Scientific Research Funds of Huaqiao University under Grant 19BS307, and the Open Project Program of Wuhan National Laboratory for Optoelectronics under Grant 2018WNLOKF009.

## References

- [1] N. Mohamed and J. Al-Jaroodi, "The impact of industry 4.0 on healthcare system engineering," in *Proceedings of the 2019 IEEE International Systems Conference (SysCon)*, pp. 1–7, Orlando, FL, USA, April 2019.
- [2] W. Ye, J. Wang, H. Tian, and H. Quan, "Public auditing for real-time medical sensor data in cloud-assisted HealthIoT system," *Frontiers of Optoelectronics*, vol. 15, no. 1, pp. 29–14, 2022.
- [3] G. Yang, Z. Pang, M. Jamal Deen et al., "Homecare robotic systems for healthcare 4.0: visions and enabling technologies," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 9, pp. 2535–2549, 2020.
- [4] A. S. Voundi Koe, Q. Chen, J. Tang et al., "Outsourcing multiauthority access control revocation and computations

- over medical data to mobile cloud,” *International Journal of Intelligent Systems*, vol. 37, no. 11, pp. 9774–9797, 2022.
- [5] K. Cresswell, R. Williams, and A. Sheikh, “Using cloud technology in health care during the COVID-19 pandemic,” *The Lancet Digital Health*, vol. 3, no. 1, pp. e4–e5, 2021.
  - [6] M. Whaiduzzaman, M. R. Hossain, A. R. Shovon et al., “A privacy-preserving mobile and fog computing framework to trace and prevent COVID-19 community transmission,” *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 12, pp. 3564–3575, 2020.
  - [7] K. Sudheep and S. Joseph, “Review on securing medical big data in healthcare cloud,” in *Proceedings of the 2019 5th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pp. 212–215, Coimbatore, India, March 2019.
  - [8] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, “Enabling public verifiability and data dynamics for storage security in cloud computing,” in *Proceedings of the European symposium on research in computer security*, pp. 355–370, Springer, Berlin, Heidelberg, September 2009.
  - [9] Y. Zhu, G. Ahn, H. Hu, S. S. Yau, H. G. An, and Chang-Jun Hu, “Dynamic audit services for outsourced storages in clouds,” *IEEE Transactions on Services Computing*, vol. 6, no. 2, pp. 227–238, 2013.
  - [10] H. Chen, H. Zhou, J. Yu et al., “Trusted audit with untrusted auditors: a decentralized data integrity crowd auditing approach based on blockchain,” *International Journal of Intelligent Systems*, vol. 36, no. 11, pp. 6213–6239, 2021.
  - [11] M. Miao, J. Li, Y. Wang, J. Wei, and X. Li, “Verifiable data streaming protocol supporting update history queries,” *International Journal of Intelligent Systems*, vol. 37, no. 12, pp. 11342–11361, 2022.
  - [12] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, “Privacy-preserving public auditing for secure cloud storage,” *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.
  - [13] B. Wang, B. Li, and H. Li, “Panda: public auditing for shared data with efficient user revocation in the cloud,” *IEEE Transactions on Services Computing*, vol. 8, no. 1, pp. 92–106, 2015.
  - [14] B. Wang, B. Li, and H. Li, “Oruta: privacy-preserving public auditing for shared data in the cloud,” *IEEE Transactions on Cloud Computing*, vol. 2, no. 1, pp. 43–56, 2014.
  - [15] C. C. Erway, A. K p c , C. Papamanthou, and R. Tamassia, “Dynamic provable data possession,” *ACM Transactions on Information and System Security*, vol. 17, no. 4, pp. 1–29, 2015.
  - [16] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, “MR-PDP: multiple-replica provable data possession,” in *Proceedings of the 28th International Conference on Distributed Computing Systems*, pp. 411–420, Beijing, China, June 2008.
  - [17] H. Yu, Z. Yang, M. Waqas et al., “Efficient dynamic multi-replica auditing for the cloud with geographic location,” *Future Generation Computer Systems*, vol. 125, pp. 285–298, 2021.
  - [18] Y. Yu, M. H. Au, G. Ateniese et al., “Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage,” *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 767–778, 2017.
  - [19] Y. Wang, Q. Wu, B. Qin, W. Shi, R. H. Deng, and J. Hu, “Identity-based data outsourcing with comprehensive auditing in clouds,” *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 940–952, 2017.
  - [20] R. Ding, H. Zhong, J. Ma, X. Liu, and J. Ning, “Lightweight privacy-preserving identity-based verifiable IoT-based health storage system,” *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8393–8405, 2019.
  - [21] B. Wang, B. Li, H. Li, and F. Li, “Certificateless public auditing for data integrity in the cloud,” in *Proceedings of the 2013 IEEE Conference on Communications and Network Security (CNS)*, pp. 136–144, National Harbor, MD, USA, October 2013.
  - [22] H. Yang, S. Jiang, W. Shen, and Z. Lei, “Certificateless provable group shared data possession with comprehensive privacy preservation for cloud storage,” *Future Internet*, vol. 10, no. 6, pp. 49–65, 2018.
  - [23] J. Li, H. Yan, and Y. Zhang, “Certificateless public integrity checking of group shared data on cloud storage,” *IEEE Transactions on Services Computing*, vol. 14, no. 1, pp. 71–81, 2021.
  - [24] J. R. Gudeme, S. Pasupuleti, and R. Kandukuri, “Certificateless privacy preserving public auditing for dynamic shared data with group user revocation in cloud storage,” *Journal of Parallel and Distributed Computing*, vol. 156, pp. 163–175, 2021.
  - [25] J. R. Gudeme, S. K. Pasupuleti, and R. Kandukuri, “Certificateless multi-replica public integrity auditing scheme for dynamic shared data in cloud storage,” *Computers and Security*, vol. 103, pp. 102176–102193, 2021.
  - [26] Z. Xu, D. He, P. Vijayakumar, B. Gupta, and J. Shen, “Certificateless public auditing scheme with data privacy and dynamics in group user model of cloud-assisted medical WSNs,” *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 5, pp. 2334–2344, 2023.
  - [27] J. Zhou, Z. Cao, X. Dong, and X. Lin, “PPDM: a privacy-preserving protocol for cloud-assisted e-healthcare systems,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 7, pp. 1332–1344, 2015.
  - [28] S. Roy, A. K. Das, S. Chatterjee, N. Kumar, S. Chattopadhyay, and J. J. Rodrigues, “Provably secure fine-grained data access control over multiple cloud servers in mobile cloud computing based healthcare applications,” *IEEE Transactions on Industrial Informatics*, vol. 15, no. 1, pp. 457–468, 2019.
  - [29] A. Juels and B. S. Kaliski Jr, “PORS: proofs of retrievability for large files,” in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 584–597, Alexandria, VA, USA, October 2007.
  - [30] G. Ateniese, R. Burns, R. Curtmola et al., “Provable data possession at untrusted stores,” in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 598–609, Alexandria, VA, USA, October 2007.
  - [31] J. L. Hernandez-Ardieta, A. I. Gonzalez-Tablas, B. Ramos, and A. Ribagorda, “Extended electronic signature policies,” in *Proceedings of the 2nd International conference on Security of information and networks*, pp. 268–277, Famagusta, North Cyprus, October 2009.
  - [32] J. A. Akinyele, C. Garman, I. Miers et al., “Charm: a framework for rapidly prototyping cryptosystems,” *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.