A provably secure certificateless signature scheme with anonymity for Healthcare IIoT

Ran Xu, Yanwei Zhou, Bo Yang

Abstract—**The industrial Internet of Things (IIoT) is changing our way of life and work. As the number of mobile devices connected to IIoT increases, users will face many security challenges such as insecure communication environments. The certificateless signature (CLS) scheme can ensure the integrity and validity of data and provide secure identity authentication for the IIoT, and several pairing-free CLS schemes have been proposed in recent years. However, we find they are vulnerable to the signature forgery attack of malicious key generation centers by safety analysis, which does not realize the security they have claimed. To solve this problem, we show an improved CLS scheme that achieves anonymity and formally proves its security under the hardness of the discrete logarithm problem in the random oracle. Comprehensive performance analysis and comparison show that our scheme has less communication and computation costs with higher security, and our construction is suitable for scenarios with limited resources. Finally, we apply this scheme to construct a mutual identity authentication protocol in the healthcare IIoT environment.**

Index Terms—**Certificateless signature; Authentication; Provable security; Healthcare industrial internet of things.**

I. Introduction

In recent years, the success of the Internet of Things (IoT) has recently spread to the industrial sector, connecting previously isolated components and the Internet through an open and global industrial network platform. This trend is commonly known as the Industrial Internet of Things (IIoT). IIoT with its characteristics of comprehensive perception, intelligent processing, self-organization, and self-maintenance, has caused a wave of change "represented by the Industrial Internet of Things" around the world [1]. IIoT has greatly improved the production efficiency of industrial agriculture, medical and health care, etc [2], [3]. In the medical field, the healthcare industrial internet of things (healthcare IIoT) has greatly promoted the intelligence and informatization of the medical system by monitoring and storing medical information and providing timely and effective medical diagnosis services [4]. The medical research team obtains health data such as blood oxygen, blood pressure, heart rate by developing micro medical sensors and placing sensor nodes on the patient's body [5]. Since devices have limited computing and storage capabilities, and the sensor data is growing exponentially, the cloud server is needed to store data. However, there are serious security problems in data outsourcing, patients' health data are vulnerable to malicious attacks during transmission and sharing [6]. The disclosure and forgery of health information damage patients' privacy, and even affect the diagnosis results of doctors, threatening life health, and leading to the deterioration of doctor-patient relationships. Therefore, it is very significant to ensure the validity and security of data transmission. Digital signatures can provide identity authentication and protect the confidentiality and integrity of data. The medical sensor collects the patient's health information, intelligent devices upload the collected information to the cloud server, and data transmission security is guaranteed through digital signature. The doctor obtains the message to remotely monitor the patient's body, and carries out the timely diagnosis, then transfers the temporary treatment plan to the patient to ensure the life health of the patient. At the same time, to protect the privacy of doctors and patients, and ensure that even if attackers can obtain some information, they cannot know the real identity of the communication entity, anonymity should also be guaranteed during data transmission.

Manuscript received December 1, 2012; revised August 26, 2015. This work is supported by the National Natural Science Foundation of China (62272287), the Key Research and Development Project in Shaanxi Province (2024GX-YBXM-074), and the Fundamental Research Funds for the Central Universities (GK202301009). Corresponding author: Y. Zhou (Email: zyw@snnu.edu.cn).

Y. Zhou is with the School of Computer Science, Shaanxi Normal University, Xi'an, China; with the State Key Laboratory of Cryptology, P. O. Box 5159, Beijing, 100878, China.

R. Xu and B. Yang are with the School of Computer Science, Shaanxi Normal University, Xi'an, China.

Certificate management problem has existed in traditional public key infrastructure. Although this problem can be solved by identity-based public key cryptography , there is an inherent key escrow problem in the this system. A cryptosystem called the certicateless public key cryptography (CL-PKC) is proposed in [7]. In CL-PKC, the user's complete private key is generated by user and the key generation center (KGC) where user chooses the secret value and KGC generates the user's partial private key. This cryptosystem does not need digital certificates and overcomes certificate management problem . In addition, KGC only has part of the private key, which solves the key escrow problem. To further describe the security, Girault [8] defined three security levels for a trusted third party, where level 3 means the strongest security for the system. In [9], Chen *et al.* proposed three types of adversaries in the CLS scheme and proved their construction has achieved Giraults level 3 security. Karati *et al.* [10] reached Giraults level 3 security in their certificateless encryption scheme.

A. Related Works

Because bilinear mapping is time-consuming [11]–[13], some pairing-free certificateless signature (CLS) schemes were proposed to improve computational efficiency. Jia *et al.* [14] created a construction of CLS scheme, but Du *et al.* [15] and Xiang *et al.* [16] proved that it is vulnerable to Type-I adversary. Ma *et al.* [17] found the above two schemes [15], [16] are both easily attacked by Type-I adversaries. Wang *et al.* [18] proposed a CLS scheme for resource-limited systems, and the computation cost is too high. Thumbur *et al.* [19] showed a CLS scheme and proved its unforgeability. Xu *et al.* [20] noted that the above scheme will be broken by the forgery attack of Type-I adversaries. They constructed a new scheme and then proved the security of this scheme. Recently, Wang *et al.* [21] illustrated Xu *et al.*'s scheme [20] is vulnerable to KGC compromised and replay attacks. They introduced a decentralized smart contract executing on the Ethereum platform to replace the traditional centralized KGC. Unfortunately, we find Wang *et al.*s scheme [21] cannot resist the attack of Type-II adversary and does not achieve the intended purpose (the detailed security analysis is given in Section III). Liu *et al.* [22] proposed a new certificateless blind signature scheme achieving conditional revocation function. Kyung-Ah Shim [23] showed a new secure CLS scheme for cloud-assisted Industrial Internet of Things. However, the above CLS schemes only achieved Girault's level 2 security, Shim *et al.*'s scheme [24] and Yang *et al.*'s scheme [25] have achieved the Girault's level 3 security, but they didn't give a specific security proof.

Therefore, we propose a new CLS scheme and prove that it can resist the attack of Type-I, Type-II and Type-III adversaries achieving the Girault's level 3 security. Through performance analysis, it is shown that our CLS scheme has lower communication and computing costs. Based on all the above analyses, our scheme can apply to lightweight health IIoT devices.

As discussed above, in the past few years, many CLS schemes [18], [20], [21], [26]–[28] have been proposed. However, some of these are based on bilinear mapping and have low computational efficiency, and some have security defects and do not meet the security requirements.

B. Our Motivations

Without loss of generality, researchers create some CLS schemes to ensure the validity of communication data. However, from the above description, we can find that many constructions do not meet the required security requirements for signature forgery attacks. In this paper, to further solve the above problems, we provide the correct method for building a secure CLS solution to avoid such security issues in the future.

In health IIoT, when doctors and patients send messages to each other, they need to confirm the legitimacy of each other's identities, so a mutual identity authentication protocol should be employed to ensure that both parties are legitimate entities. At the same time, to protect privacy in the communication process, the corresponding protocol has realized anonymity. Furthermore, the devices typically have limited computing and storage capabilities, and the corresponding protocol needs to provide highly efficient computing and communication, and we can use the proposal in resource-constrained environments.

As discussed above, we should construct a secure CLS scheme and design efficient authentication protocols based on our proposal.

C. Our Contributions

In this paper, we focus on the security of data transmission in healthcare IIoT by designing a provably secure CLS scheme and constructing a mutual identity authentication protocol in the medical environment based on our CLS scheme, which protects privacy information for both sides of the communication through anonymity. We summarize the main specific contributions as the following:

For Wang *et al.*s CLS scheme [21], we first point out some computational problems of an additive group. After that, we analyze the security to show that the Type-II adversary can perform signature forgery attacks.

We show a novel CLS scheme without pairing operations and then prove it has achieved Giraults level 3 security. That is, our proposal has attached the highest security level. Furthermore, a comprehensive comparison of the corresponding CLS schemes is employed to prove that the above construction has less communication and computation costs with higher security and also shows that it is suitable for resource-constrained scenarios of health IIoT.

Finally, we propose a mutual identity authentication and key agreement protocol utilizing our CLS scheme, which realizes identity authentication between doctors and patients in the actual Health IIoT environment and proves that the above protocol has better security and performance. We conduct simulation experiments to integrate our scheme with the healthcare system, demonstrating our protocol applied in the healthcare system by experiment.

II. Preliminaries

The symbols used in this paper are shown in Table I.

TABLE I
THE LIST OF SYMBOL DEFINITION

Symbol	Definition
G	addition cyclic group
Z_q^*	finite field
κ	security parameter
msk	master secret key
$params$	system public parameter
d	partial private key
RID	the real identity of user
ID	the temporary identity of user
PK	public key
SK	private key
$\mathcal{A}_1$	Type I adversary
$\mathcal{A}_2$	Type II adversary
$\mathcal{A}_3$	Type III adversary

A. Girault's Security Level

In [8], Girault distinguished the authority (e.g., KGC) into three trust levels. Then, Hu *et al.* [29] extended the above security level from the CLS scheme. The three levels are as follows:

Level 1. KGC masters users' private keys, who can forge the signature as an impersonated user.

Level 2. KGC does not know users' private keys, and it also can forge the signature by forging the false private key as an impersonated user.

Level 3. KGC does not know users' private keys, who cannot forge the signature by forging a false private key. The forged private keys for all users cannot be undetected.

B. Complexity Assumption

Definition 1 (Discrete Logarithm (DL) Assumption): Let G be an addition cyclic group with prime order q, and P is the generator of G. For any probabilistic polynomial time adversary, it is difficult to compute $a \in Z_q^*$ given $Q = aP$. That is, the advantage of the adversary $\mathcal{A}$ is described as $\mathsf{Adv}_{\mathcal{A}}^{DL}(\kappa) = \Pr[\mathcal{A}(P, aP) = a]$.

The DL assumption shows that, for any adversary $\mathcal{A}$, the above advantage $\mathsf{Adv}_{\mathcal{A}}^{DL}(\kappa)$ is negligible.

C. System Model

The system framework of healthcare industrial internet of things is as shown in Fig 1, next, we will provide a specific introduction.

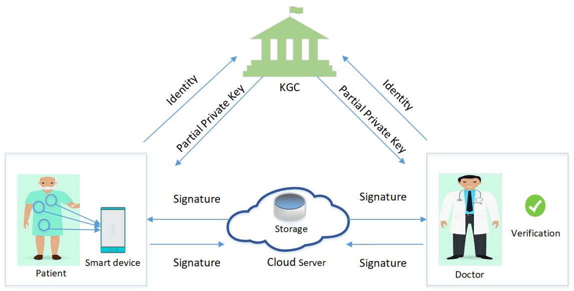


Fig. 1. System framework of healthcare industrial internet of things

1) Participating Entities: The main participating entities are the following:

(1) ***KGC***. KGC produces public parameters and master private keys for the medical system and generates partial private keys for patients and doctors. In the healthcare system, it plays the role of the hospital management department.

(2) ***Patient***. Patients' health information (e.g., blood pressure, heart rate, etc.) can be collected by smart devices and uploaded to cloud servers. Finally, patients obtain the diagnostic results from the cloud server uploaded by doctors.

(3) ***Cloud Sever***. The cloud server can store uploaded body data from patients and uploaded diagnostic results from doctors. The cloud server provides data storage functions for the healthcare IIoT, and provides security protection for data, reducing the pressure of data storage for users. In particular, this paper no longer discusses the secure storage of data on cloud servers, which is a new topic.

(4) ***Doctor***. Doctors can download patients' health information from the cloud server, diagnose diseases, and then upload the diagnostic results to the cloud server.

2) Security Threats: The following are potential security issues in the system.

(1) ***Insider Attacks***. The third-party KGC may be corrupt, and internal attacks can be executed.

(2) ***Impersonation Attacks***. There may be malicious third-party impersonating users or corrupt internal users, which can cause an external attack.

(3) ***Replay Attacks***. Malicious users may disrupt the correctness of the authentication process through replay.

(4) ***Man-in-the-middle (MITM) attacks***. During the communication process between patients and doctors, malicious attackers manipulate the communication process, where the attacker obtains and modifies the message and then passes it on.

(5) ***Data integrity attacks***. Attackers can maliciously tamper with the transmitted message, damaging the integrity of the message.

(6) ***Data confidentiality attacks***. Attackers grab private messages during the communication process, posing a threat to the confidentiality of data.

Based on this, we will propose a CLS scheme and apply it in the mutual identity authentication protocol in the healthcare system proving our scheme can resist the above attack and protect the legitimacy and security of data.

D. Certificateless Signature Scheme

Now, we will review the formal definition and security model of the CLS scheme.

1) Formal Definition: A CLS scheme has the following six algorithms.

Setup. KGC executes the setup algorithm given the security parameter κ and produces master secret key msk and parameters $params$. That is, $(params, msk) \leftarrow \mathsf{Setup}(1^\kappa)$.

Partial Private Key. KGC inputs the identity ID, msk and parameters $params$, and runs the algorithm, finally, KGC outputs the d as the partial private key to the user. That is, $d \leftarrow \mathsf{ParPriKey}(ID, params, msk)$.

Secret value. User executes this algorithm given the parameters $params$ and ID, then returns x as the secret value. That is, $x \leftarrow \mathsf{SecVal}(params, ID)$.

Public/Private Key. User executes this algorithm given the parameters $params$, ID, d, and x as input, then outputs the public key and private key (PK, SK), respectively. That is, $(PK, SK) \leftarrow \mathsf{KeyGen}(params, ID, d, x)$.

Signature. The signer inputs parameters $params$, message m, identity ID, and private key SK, then runs this algorithm and outputs the signature δ about the message m. That is, $\delta \leftarrow \mathsf{Sign}(params, ID, SK, m)$.

Verify. The verifier takes the parameters $params$, ID, m, and δ as input, and runs the algorithm. Output "1" if the δ is valid; Otherwise, output "0". That is, $1/0 \leftarrow \mathsf{Verify}(params, ID, m, \delta)$.

2) Security Model: According to the different types of attacks on CLS schemes, we classify the adversaries into three types: Type I adversaries $\mathcal{A}_1$, Type II adversaries $\mathcal{A}_2$ and Type III adversaries $\mathcal{A}_3$. $\mathcal{A}_1$ initiates an external attack, which can perform public-key replacement attacks. However, $\mathcal{A}_1$ can not capture the user's partial private key and the master secret key. $\mathcal{A}_2$ initiates an internal attack, which is a malicious KGC and can start a master secret key attack but cannot replace the public key of users. $\mathcal{A}_3$ impersonates a legitimate user, which forges public and private keys that don't hold by itself.

The unforgeability of the CLS scheme requires that the corresponding scheme is unforgeable under the chosen message attack, and we define two games as the following to prove it.

Game 1: This game is executed between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}_1$, the specific execution process is as follows:

(1) Initialization. $\mathcal{C}$ executes the setup algorithm Setup, and produces master secret key msk which is kept secretly and parameters $params$, then sends $params$ to $\mathcal{A}_1$.

(2) Queries. At this stage, $\mathcal{A}_1$ can perform the following inquiries of polynomial bounded degree. Furthermore, the following queries are performed adaptively.

- $Creation\text{-}User$: When $\mathcal{A}_1$ asks the creation user query with the identity ID, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ to create d, runs SecVal to create x, runs KeyGen to create (PK, SK), and returns PK to $\mathcal{A}_1$.
- $Extract\text{-}Partial\text{-}Private\text{-}Key$: When $\mathcal{A}_1$ asks this query with ID, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ and returns d to $\mathcal{A}_1$.
- $Extract\text{-}Secret\text{-}Value$: When $\mathcal{A}_1$ asks this query with ID, $\mathcal{C}$ executes SecVal, and returns x to $\mathcal{A}_1$.
- $Extract\text{-}Private\text{-}Key$: When $\mathcal{A}_1$ asks this query with ID,, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ to create d, runs SecVal to create x, runs KeyGen to create (PK, SK), and returns SK to $\mathcal{A}_1$.
- $Replace\text{-}Public\text{-}Key$: When $\mathcal{C}$ receiving this query, use a new value PK^* to replace public key PK.
- $Signature$: When $\mathcal{A}_1$ asks this query with (ID, m), $\mathcal{C}$ executes signature algorithm Sign to generate δ, and outputs it to $\mathcal{A}_1$.

(3) Forgery. $\mathcal{A}_1$ outputs a forgery signature δ^* for the challenge identity message pair (ID^*, m^*), when the following conditions are all met, the adversary $\mathcal{A}_1$ wins the game:

- $\mathcal{A}_1$ cannot ask $Extract\text{-}Partial\text{-}Private\text{-}Key$ and $Extract\text{-}Private\text{-}Key$ queries for ID^*.
- For any identity of the replaced public key, $\mathcal{A}_1$ can not perform $Extract\text{-}Private\text{-}Key$ query.
- $\mathcal{A}_1$ can never ask a signature query for (ID^*, m^*), and δ^* is a valid signature.

Therefore, for any PPT adversary $\mathcal{A}_1$, the advantage of winning this game is $\mathsf{Adv}_{\mathcal{A}_1}(\kappa) = \Pr[\mathcal{A}_1\ wins]$.

Game 2: This game is executed between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}_2$, the detailed execution process is following:

(1) Initialization. $\mathcal{C}$ executes the setup algorithm Setup, and produces master secret key msk which is kept secretly and parameters $params$, then sends $params$ to $\mathcal{A}_2$.

(2) Queries. At this stage, $\mathcal{A}_2$ can conduct the following inquiries of polynomial bounded degree. Furthermore, the following queries are performed adaptively.

- $Creation\text{-}User$: When $\mathcal{A}_2$ asks the creation user query with the identity ID, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ to create d, runs SecVal to create x, runs KeyGen to create (PK, SK), and returns PK to $\mathcal{A}_2$.
- $Extract\text{-}Secret\text{-}Value$: : When $\mathcal{A}_2$ asks this query with ID, $\mathcal{C}$ executes SecVal, and returns x to $\mathcal{A}_2$.
- $Extract\text{-}Private\text{-}Key$: When $\mathcal{A}_2$ asks this query with ID, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ to create d, runs SecVal to create x, runs KeyGen to create (PK, SK), and returns SK to $\mathcal{A}_2$.
- $Signature$: When $\mathcal{A}_2$ asks this query with (ID, m), $\mathcal{C}$ executes signature algorithm Sign to generate δ, and returns it to $\mathcal{A}_2$.

(3) Forgery. $\mathcal{A}_2$ outputs a forgery signature δ^* for the challenge identity message pair (ID^*, m^*), when the following conditions are all met, the $\mathcal{A}_2$ wins this game:

- At any stage, $\mathcal{A}_2$ can not ask $Extract\text{-}Private\text{-}Key$ query for the identity ID^*.
- $\mathcal{A}_2$ never ask the signature query for (ID^*, m^*), and δ^* is a valid signature.

Therefore, for any PPT adversary $\mathcal{A}_2$, the advantage of winning this game is $\mathsf{Adv}_{\mathcal{A}_2}(\kappa) = \Pr[\mathcal{A}_2\ wins]$.

Definition 2 (Unforgeability): For PPT adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$, if the corresponding advantages $\mathsf{Adv}_{\mathcal{A}_1}(\kappa)$ and $\mathsf{Adv}_{\mathcal{A}_2}(\kappa)$ are negligible in the above games, then, the corresponding CLS scheme has unforgeability.

The non-repudiation of the CLS scheme requires that the corresponding scheme is undeniable under the chosen message attack, and we define the following game to prove it.

Game 3: This game is executed between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}_3$, the specific execution process is as follows:

(1) Initialization. $\mathcal{C}$ executes the setup algorithm Setup, and produces master secret key msk which is kept secretly and parameters $params$, then sends $params$ to $\mathcal{A}_1$.

(2) Queries. At this stage, $\mathcal{A}_3$ can conduct the following inquiries of polynomial bounded degree. Furthermore, the following queries are performed adaptively.

- $Creation\text{-}User$: When $\mathcal{A}_1$ asks the creation user query with the identity ID, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ to create d, runs SecVal to create x, runs KeyGen to create (PK, SK), and returns PK to $\mathcal{A}_3$.
- $Extract\text{-}Partial\text{-}Private\text{-}Key$: When $\mathcal{A}_3$ asks this query with ID, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ and returns d to $\mathcal{A}_3$.
- $Extract\text{-}Secret\text{-}Value$: When $\mathcal{A}_3$ asks this query with ID, $\mathcal{C}$ executes SecVal, and returns x to $\mathcal{A}_3$.
- $Extract\text{-}Private\text{-}Key$: When $\mathcal{A}_3$ asks this query with the identity ID, $\mathcal{C}$ executes $\mathsf{ParPriKey}$ to create d, runs SecVal to create x, runs KeyGen to create (PK, SK), and returns SK to $\mathcal{A}_3$.
- $Signature$: When $\mathcal{A}_3$ asks this query with the identity message pair (ID, m), $\mathcal{C}$ executes signature algorithm Sign to generate δ, and outputs it to $\mathcal{A}_3$.

(3) Forgery. $\mathcal{A}_3$ outputs a forged public-private key pair (PK^*, SK^*) for the challenge identity ID^*, when the following conditions are all met, the adversary $\mathcal{A}_3$ wins the game:

(1) $\mathcal{A}_3$ cannot ask $Extract\text{-}Partial\text{-}Private\text{-}Key$ and $Extract\text{-}Private\text{-}Key$ queries for ID^*.

(2) The forged public-private key pair (PK^*, SK^*) is valid and different from the pair that is created from creation user query for ID^*.

Therefore, for any PPT adversary $\mathcal{A}_3$, the advantage of winning this game is $\mathsf{Adv}_{\mathcal{A}_3}(\kappa) = \Pr[\mathcal{A}_3\ wins]$.

Definition 3 (Non-repudiation): For PPT adversary $\mathcal{A}_3$, if the corresponding advantages $\mathsf{Adv}_{\mathcal{A}_3}(\kappa)$ is negligible in the above games, then, the corresponding CLS scheme has non-repudiation.

III. Cryptanalysis of Wang et al.'s CLS scheme

Wang *et al.*'s scheme [21] includes six algorithms, we review them as follows:

Setup. Input a security parameter, KGC selects a group G with the order q and the generator is P. Computes $P_{pub} = sP$, sets $s \in Z_q^*$ as system master key. Let $h_1, h_2 : \{0,1\}^* \to Z_q^*$ be two hash functions. Finally, KGC sets public parameters is $params = \{G, P, P_{pub}, q, h_1, h_2\}$.

Partial Private Key. This algorithm consists of the following operations:

The user U_i computes $P_{ID_i} = ID_iP$, $C_{ID_i} = ID_i + ID_iP_{pub}$, where $ID_i \in \{0,1\}^*$, and sends request message (P_{ID_i}, C_{ID_i}) to KGC.

If $C_{ID_i} - sP_{ID_i} = ID_i$, then KGC selects $r_i \in Z_q^*$, and calculates $R_i = r_iP$, $ppk_i = r_i + s\alpha_i \bmod q$, where $\alpha_i = h_1(P_{pub}, ID_i, R_i)$. Next, KGC returns ppk_i to user U_i.

Public/Private Key. The user U_i chooses $x_i \in Z_q^*$ as secret value, and calculates $X_i = x_iP$. Finally, set public key as $PK_i = (X_i, R_i)$ and private key as $SK_i = (x_i, ppk_i)$.

Signature. For the message $m_i \in \{0,1\}^*$, the user U_i computes $\beta_i = h_2(P_{pub}, ID_i, PK_i, m_i)$ and $sig_i = (ppk_i - \beta_ix_i) \bmod q$. Finally, sends $\delta_i = (PK_i, sig_i)$ to the verier V_j.

Verication. For $\beta_i = h_2(P_{pub}, ID_i, PK_i, m_i)$ and $\alpha_i = h_1(P_{pub}, ID_i, R_i)$, V_j will verify if $sig_iP = R_i + \alpha_iP_{pub} - \beta_iX_i$ holds.

Notice that, through our analysis, we found that the above CLS scheme has the following shortcomings.

(1) Wang *et al.* set $ID_i \in \{0,1\}^*$, but there is no definition for the operation of multiplying the string by the generator in the addition group, so it is wrong. That is, ID_iP and ID_iP_{pub} are incalculable.

(2) We find that Wang *et al.*'s CLS scheme [21] is vulnerable for the Type-II adversary. In the signature generation algorithm, U_i calculates $sig_i = (ppk_i - \beta_ix_i) \bmod q$. Because the Type II adversary is a curious and honest KGC, it can get the partial private key ppk_i. Furthermore, sig_i and β are public. In this case, x_i can be obtained by calculating $x_i = \frac{ppk_i - sig_i}{\beta_i}$. Therefore, the Type-II adversary can obtain the private key SK_i and will forge signatures at will.

(3) The signature operation of this CLS scheme does not use a random number, which is not a randomized signature. Therefore, a Type II adversary can forge a valid signature using the existing signature. The scheme is insecure, and it is not suitable for real-world application.

Therefore, to further solve the above problems to achieve a secure proposal, a new construction of the CLS scheme is created in this paper.

IV. Our Novel Construction of CLS Scheme

Our improved CLS scheme with better security is created in this section.

A. Concrete Construction

The six algorithms of our CLS scheme are illustrated as follows:

(1) **Setup**. KGC does the following operations:

Input a security parameter κ, choose an additive group G with order q and the generator is P. Choose $s \in Z_q^*$ as the system master secret key and compute $P_{pub} = sP$. Define four one-way hash functions $H_0 : G \times \{0,1\}^* \to \{0,1\}^*$, $H_1, H_2, H_3, H_4 : \{0,1\}^* \to Z_q^*$. For convenience, we define the corresponding input for the above four hash functions as strings with arbitrarily long. Finally, set $Params = \{G, q, P, P_{pub}, H_0, H_1, H_2, H_3, H_4\}$ as system public parameters.

(2) **Partial Private Key Generation**. The user U_i chooses $x \in Z_q^*$ as secret value and then calculates $X = xP$. Afterward, U_i sends the real identity RID and the public parameter X to KGC. After receiving RID and X from U_i, KGC selects $r \in Z_q^*$, then computes the pseudonym $ID = RID \oplus H_0(rP_{pub}, V_t)$, where V_t is the validity period of ID. Next, KGC calculates $d = (r + sh_1) \bmod q$ and $R = rP$, where $h_1 = H_1(P_{pub}, ID, R, X)$. Finally, returns (d, R) as the partial key.

Notice that, in our construct, the real identity is replaced by a temporary identity generated by KGC, providing anonymity for the user. In addition, anonymity is controllable, as KGC can reveal the malicious real identity. Our CLS scheme achieves anonymity by hiding real identity and using pseudonyms to protect their privacy.

(3) **Key Generation**. User U_i verifies the effectiveness of pseudonym ID and the correctness of partial key (d, R) by calculating $dP = R + h_1P_{pub}$. Afterward, U_i sets $SK = (x, d)$ as the the private key, and $PK = (X, R)$ as the public key.

(4) **Signature**. Let $Time$ be a time stamp. For the message $m \in \{0,1\}^*$, the signer U_i randomly selects $t \in Z_q^*$, compute $\sigma = h_4t + h_3d + h_2x$ and $T = tP$ where $h_2 = H_2(P_{pub}, ID, PK, m\|Time, T)$, $h_3 = H_3(P_{pub}, ID, PK, m\|Time, T)$ and $h_4 = H_4(P_{pub}, ID, PK, m\|Time, T)$. Finally, U_i outputs $\delta = (T, \sigma)$ as signature and uploads it to cloud server.

(5) **Verify**. Verfier obtains δ from cloud server. First check times tamp $Time$, if it is valid, then continue the verification process. For the signature $\delta = (T, \sigma)$ and the message m, the verfier compute $h_1 = H_1(P_{pub}, ID, R, X)$, $h_2 = H_2(P_{pub}, ID, m\|Time, PK, T)$, $h_3 = H_3(P_{pub}, ID, m\|Time, PK, T)$ and $h_4 = H_4(P_{pub}, ID, PK, m\|Time, T)$, if $\sigma P = h_4T + h_3(R + h_1P_{pub}) + h_2X$ holds, then outputs "1"; Otherwise, verifier outputs "0".

B. Correctness

Notice that, the correctness of our proposal can be achieved from the equations as follows.

$$\begin{aligned} dP &= (r + sh_1)P = R + h_1P_{pub} \\ \sigma P &= (h_4t + h_3d + h_2x)P \\ &= h_4T + h_3(R + h_1P_{pub}) + h_2X. \end{aligned}$$

C. Security Proof

In this part, we prove our scheme has better security by demonstrating it can resist the attack of Type-I adversary $\mathcal{A}_1$, Type-II adversary $\mathcal{A}_2$ and Type-III adversary $\mathcal{A}_3$, the detailed poof process is the following.

Theorem 1: If there exists a Type-I adversary $\mathcal{A}_1$ who can break the unforgeability of the CLS scheme with obvious advantage ε_1, then there exists a challenger $\mathcal{C}$ who can solve the hardness of DL assumption with non-negligible advantage $(1-\frac{1}{e})\frac{\varepsilon_1}{e(Q_1+Q_2+1)Q_h}$. We denote the queries times of H_1 oracle, partial private key generation and private key generation as Q_h, Q_1, Q_2, and e is the natural log base.

Proof 1: Let $\mathcal{A}_1$ be a Type-I adversary who can forge the signature with obvious advantage ε_1. Now, we construct a challenger $\mathcal{C}$ who can solve the hardness of DL assumption by using $\mathcal{A}_1$. In the beginning, a challenge tuple $(P, Q = sP)$ of DL assumption should be obtained by $\mathcal{C}$, and the goal is to find $s \in Z_q^*$. The game between $\mathcal{C}$ and $\mathcal{A}_1$ is the following: **(1) Initialize**. $\mathcal{C}$ sets $P_{pub} = Q = sP$, and initializes five lists L_1, L_2, L_3, L_4 and L_u used to record the queried message. Finally, $\mathcal{C}$ returns $Params = \{G, q, P, P_{pub}, H_0, H_1, H_2, H_3, H_4\}$ to $\mathcal{A}_1$, where H_1, H_2 and H_3 are three random oracles.

(2) Queries. $\mathcal{A}_1$ adaptively asks the queries as follows at this stage. The challenger $\mathcal{C}$ chooses a random identity ID^* from the space submitted by $\mathcal{A}_1$ as the challenge identity. Specially, from here on, the parameters related to the challenging identity ID^* are marked with "$*$".

$Creation\ User$. When $\mathcal{A}_1$ submits the query of creation user with identity ID as input, if $(ID, X, R, x, d) \in L_u$, $\mathcal{C}$ outputs the public key $PK = (X, R)$ to $\mathcal{A}_1$; Otherwise, $\mathcal{C}$ selects random values $d, x, h_1 \in Z_q^*$, calculates $R = dP - h_1P_{pub}$ and $X = xP$, then adds (ID, R, h_1) into the list L_1 and (ID, X, R, x, d) into the list L_u. Finally, $\mathcal{C}$ returns $PK = (X, R)$ to $\mathcal{A}_1$. Notice that, for d, there exists a random value $r \in Z_q^*$ that satisfies $d = r + sh_1$.

$Partial\ Private\ Key$. When $\mathcal{A}_1$ submits the partial private key generation query with inputting ID, $\mathcal{C}$ will abort if $ID = ID^*$; Otherwise, $\mathcal{C}$ uses the ID as index to seek a tuple (ID, X, R, x, d) from L_u , if it exists, $\mathcal{C}$ returns (d, R) to $\mathcal{A}_1$; else, $\mathcal{C}$ runs the creation user query for ID and outputs (d, R) to $\mathcal{A}_1$.

$Secret\ Value$. When $\mathcal{A}_1$ submits a secret value generation query with ID as input, if $ID = ID^*$, then $\mathcal{C}$ will abort; Otherwise, $\mathcal{C}$ seeks a tuple (ID, X, R, x, d) from L_u with ID as index, if it exists, then $\mathcal{C}$ returns x to $\mathcal{A}_1$; else, $\mathcal{C}$ executes the query of creation user for ID and outputs x to $\mathcal{A}_1$.

$Private\ Key$. When $\mathcal{A}_1$ submits the query of private key generation with inputting ID, if $ID = ID^*$, $\mathcal{C}$ will abort; Otherwise, $\mathcal{C}$ make ID as index to seeks the tuple (ID, X, R, x, d) from L_u, if it exists, $\mathcal{C}$ returns $SK = (x, d)$ to $\mathcal{A}_1$; else, $\mathcal{C}$ executes the creation user query for ID and returns $SK = (x, d)$ to $\mathcal{A}_1$.

$H_1\ Query$. When $\mathcal{A}_1$ submits an H_1 oracle query with (ID, R, P_{pub}) as input, $\mathcal{C}$ returns h_1 to $\mathcal{A}_1$ from the list L_1 with (ID, R, P_{pub}) as an index.

$H_2\ Query$. When $\mathcal{A}_1$ submits an H_2 oracle query with the tuple (ID, PK, P_{pub}, m, T) as input, if $(ID, PK, P_{pub}, m, T, h_2) \in L_2$, then $\mathcal{C}$ outputs h_2 to $\mathcal{A}_1$; Otherwise $\mathcal{C}$ chooses a random value $h_2 \in Z_q^*$, adds $(ID, PK, P_{pub}, m, T, h_2)$ into the list L_2 and sends it to $\mathcal{A}_1$.

$H_3\ Query$. When $\mathcal{A}_1$ submits an H_3 oracle query with the tuple (ID, PK, P_{pub}, m, T) as input, if $(ID, PK, P_{pub}, m, T, h_3) \in L_3$, then $\mathcal{C}$ outputs h_3 to $\mathcal{A}_1$; Otherwise $\mathcal{C}$ chooses a random value $h_3 \in Z_q^*$, adds $(ID, PK, P_{pub}, m, T, h_3)$ into the list L_3 and sends it to $\mathcal{A}_1$.

$H_4\ Query$. When $\mathcal{A}_1$ submits an H_4 oracle query with the tuple (ID, PK, P_{pub}, m, T) as input, if $(ID, PK, P_{pub}, m, T, h_4) \in L_4$, then $\mathcal{C}$ outputs h_4 to $\mathcal{A}_1$; Otherwise $\mathcal{C}$ chooses a random value $h_4 \in Z_q^*$, adds $(ID, PK, P_{pub}, m, T, h_4)$ into the list L_4 and sends it to $\mathcal{A}_1$.

$Public\ Key\ Replacement$. When $\mathcal{A}_1$ submits a public key replacement query with $(ID, PK' = (X', R'))$ as input, $\mathcal{C}$ will ignore this query if $ID = ID^*$ holds; Otherwise, $\mathcal{C}$ seeks whether (ID, X, R, x, d) is present in the list L_u, if it exists, $\mathcal{C}$ updates it with tuple $(ID, X', R', \perp, \perp)$, else, $\mathcal{C}$ executes the query of creation user query with ID and then replaces (ID, X, R, x, d) with $(ID, X', R', \perp, \perp)$.

$Signature$. When $\mathcal{A}_1$ submits a signature generation query with inputting the identity message pair (ID, m_i), $\mathcal{C}$ performs the following operations.

1) If $ID \neq ID^*$ and $d \neq \perp$, $\mathcal{C}$ chooses $t \in Z_q^*$, then calculates $T = tP$, $h_2 = H_2(P_{pub}, ID, m, PK)$, $h_3 = H_3(P_{pub}, ID, m, PK, T)$ and $h_4 = H_4(P_{pub}, ID, m, PK, T)$. Next $\mathcal{C}$ calculates $\sigma = h_4t + h_3d + h_2x$ and outputs $\delta = (T, \sigma)$ to $\mathcal{A}_1$.

2) Otherwise, $\mathcal{C}$ selects $\sigma, h_2, h_3, h_4 \in Z_q^*$, and calculates $T = \frac{1}{h_4}\Big(\sigma P - h_3(R + h_1P_{pub}) - h_2X\Big)$. Afater that, adds $(ID, PK, P_{pub}, m_i, T, h_2)$, $(ID, PK, P_{pub}, m_i, T, h_3)$ and $(ID, PK, P_{pub}, m_i, T, h_4)$ into the lists L_2, L_3 and L_4. Finally, $\mathcal{C}$ outputs the signature $\delta = (T, \sigma)$ to $\mathcal{A}_1$. Specifically, h_1, X and R can be obtained from the corresponding lists with ID as an index.

(3) Forgery. Finally, $\mathcal{A}_1$ returns a forged signature $\delta_1^* = (T^*, \delta_1^*)$ for the challenge identity message pair (ID_i, m_i^*), where $T^* = t^*P$. If $ID_i \neq ID^*$, $\mathcal{C}$ aborts. Otherwise, according to the description of Forking lemma [30], $\mathcal{A}_1$ can forge a new signature $\delta_2^* = (T^*, \sigma_2^*)$ by the different hash response $\hat{h}_1^*$ of H_1 oracle and same random value t^*, which makes these two equations $\sigma_1^* = h_4^*t^* + h_3^*(r^* + sh_1^*) + h_2^*d^*$ and $\sigma_2^* = h_4^*t^* + h_3^*(r^* + s\hat{h}_1^*) + h_2^*d^*$ hold.

The result shows $s = \frac{\sigma_1^* - \sigma_2^*}{h_3^*(h_1^* - \hat{h}_1^*)}$, which is against the DL assumption. ξ_1 indicates that at the query stage the above game does not abort ; ξ_2 indicates that at the forgery stage the above game does not abort; ξ_3 indicates $\mathcal{A}_1$ creating two valid signatures δ_1^* and δ_2^*. Therefore, we get $\Pr[\xi_1] = \Big(1 - \frac{1}{Q_1+Q_2+1}\Big)^{Q_1+Q_2}$ and $\Pr[\xi_2] = (\frac{1}{Q_1+Q_2+1})$. Through Forking lemma, the probability of $\mathcal{A}_1$ creating two valid forged signatures δ_1^* and δ_2^* is $\Pr[\xi_3] \geq \Big(1 - \frac{1}{e}\Big)\frac{\varepsilon_1}{Q_h}$. Then, we have

$$\begin{aligned}
&\Pr[\xi_1 \wedge \xi_2 \wedge \xi_3] \\
&\geq \Big(1 - \frac{1}{Q_1+Q_2+1}\Big)^{Q_1+Q_2}\frac{1}{Q_1+Q_2+1}\Big(1-\frac{1}{e}\Big)\frac{\varepsilon_1}{Q_h} \\
&\geq \Big(1-\frac{1}{e}\Big)\frac{\varepsilon_1}{e(Q_1+Q_2+1)Q_h}.
\end{aligned}$$

In conclusion, if there exists an adversary $\mathcal{A}_1$ who can break the unforgeability of the CLS scheme with advantage ε_1, then there exists a challenger $\mathcal{C}$ who can solve the hardness of DL assumption with non-negligible probability $(1-\frac{1}{e})\frac{\varepsilon_1}{e(Q_1+Q_2+1)Q_h}$.

Theorem 2: If there exists the Type-II adversary $\mathcal{A}_2$ who can can break the unforgeability of our CLS scheme with obvious advantage ε_2, then there exists a challenger $\mathcal{C}$ who can solve the hardness of DL assumption with non-negligible advantage $(1-\frac{1}{e})\frac{\varepsilon_2}{e(Q_2+Q_3+1)Q'_h}$, where Q_3 and Q'_h are the number of secret value generation and H_2 oracle queries.

Proof 2: Let $\mathcal{A}_2$ be a TypeII adversary who can forge the valid signature with obvious advantage ε_2. Now, we construct a challenger $\mathcal{C}$ who can solve the hardness of DL assumption by using $\mathcal{A}_2$. In the beginning, a challenge tuple $(P, Q=aP)$ of DL assumption should be obtained by $\mathcal{C}$, the goal is to find $a \in Z_q^*$. The game between $\mathcal{C}$ and $\mathcal{A}_1$ is the following:

(1) Initialize. The challenger $\mathcal{C}$ chooses $s \in Z_q^*$, calculates $P_{pub} = sP$, and initializes four lists L_2, L_3, L_4 and L_u used to record the queried message. Finally, $\mathcal{C}$ returns $Params = \{G, q, P, P_{pub}, H_0, H_1, H_2, H_3, H_4\}$ to $\mathcal{A}_1$, where H_2, H_3, H_4 are three random oracles.

(2) Queries. $\mathcal{A}_2$ adaptively asks the queries as follows at this stage. The challenger $\mathcal{C}$ chooses a random identity ID^* from the space submitted by $\mathcal{A}_2$ as the challenge identity. Furthermore, $\mathcal{C}$ responds to H_2, H_3, and H_4 queries in the same way as Theorem 1.

Creation User. When $\mathcal{A}_2$ submits a query of creation user with inputting ID, if $(ID, X, R, x, d) \in L_u$, then $\mathcal{C}$ returns public key $PK = (X, R)$ to $\mathcal{A}_2$; Otherwise, $\mathcal{C}$ performs the following operations.

1) If $ID = ID^*$, then $\mathcal{C}$ makes $X^* = aP$, selects random values $r^* \in Z_q^*$, calculates $R^* = r^*P$ and $d^* = r^* + sh_1^*$, where $h_1^* = H_1(P_{pub}, ID^*, R^*)$, then adds $(ID^*, X^*, R^*, \perp, d^*)$ into L_u. Finally, $\mathcal{C}$ returns $PK^* = (X^*, R^*)$ to $\mathcal{A}_2$.

2) If $ID \neq ID^*$, then $\mathcal{C}$ selects $x, r \in Z_q^*$, calculates $X = xP$, $R = rP$ and $d = r + sh_1$, where $h_1 = H_1(P_{pub}, ID, R)$. $\mathcal{C}$ inserts the tuple (ID, X, R, x, d) in the list L_u. Finally, $\mathcal{C}$ outputs $PK = (X, R)$ to $\mathcal{A}_2$.

Secret Value. When $\mathcal{A}_2$ submits the query of secret value generation with inputting ID, if $ID = ID^*$, then $\mathcal{C}$ aborts; Otherwise, $\mathcal{C}$ seeks a tuple (ID, X, R, x, d) from L_u with ID as an index, if it exists, then $\mathcal{C}$ returns x to $\mathcal{A}_2$; else, $\mathcal{C}$ executes the creation user query for ID and returns x to $\mathcal{A}_2$.

Private Key. When $\mathcal{A}_2$ submits a private key generation query with ID as input, if $ID = ID^*$, then $\mathcal{C}$ aborts; Otherwise, $\mathcal{C}$ make ID as index to seek a tuple (ID, X, R, x, d) from L_u, if it exists, $\mathcal{C}$ outputs $SK = (x, d)$ to $\mathcal{A}_2$; else, $\mathcal{C}$ will execute the query of creation user for ID and returns $SK = (x, d)$ to $\mathcal{A}_2$.

Signature. When $\mathcal{A}_2$ submits the signature generation query with inputting (ID, m_i), $\mathcal{C}$ selects $\sigma, h_2, h_3, h_4 \in Z_q^*$, and calculates $T = \frac{1}{h_4}\Big(\sigma P - h_3(R + h_1 P_{pub}) - h_2 X\Big)$, where $h_1 = H_1(P_{pub}, ID, R)$, R and X can be obtain by performing creation user query for ID. Also, adds tuples $(ID, PK, P_{pub}, m_i, T, h_2)$, $(ID, PK, P_{pub}, m_i, T, h_3)$ and $(ID, PK, P_{pub}, m_i, T, h_4)$ into the lists L_2,L_3 and L_4. Finally, $\mathcal{C}$ returns $\delta = (T, \sigma)$ as the signature to $\mathcal{A}_2$.

(3) Forgery. Finally, $\mathcal{A}_2$ forges a signature $\delta_1^* = (T^*, \sigma_1^*)$ for the challenge identity message pair (ID_i, m_i^*). If $ID_i \neq ID^*$, then $\mathcal{C}$ aborts, and output $\perp$. Otherwise, according to Forking Lemma [30], $\mathcal{A}_2$ produces $\delta_2^* = (T^*, \sigma_2^*)$ as a new forged signature with different hash response $\hat{h}_2$ of H_2 oracle and the same random value t^*, which makes these two equations $\sigma_1^* = h_4^* t + h_3^*(r^* + sh_1^*) + h_2^* a$ and $\sigma_2^* = h_4^* t + h_3^*(r^* + sh_1^*) + \hat{h}_2^* a$ hold.

The result shows $a = \frac{\sigma_1^* - \sigma_2^*}{h_2^* - \hat{h}_2^*}$, which is against the DL assumption. $\mathcal{F}_1$ indicates that at the query stage the above game does not abort; $\mathcal{F}_2$ indicates that at the forgery stage the above game does not abort ; $\mathcal{F}_3$ indicates $\mathcal{A}_2$ creating two valid signatures δ_1^* and δ_2^*. Therefore, we get $\Pr[F_1] = \Big(1 - \frac{1}{Q_2+Q_3+1}\Big)^{Q_2+Q_3}$ and $\Pr[F_2] = \frac{1}{Q_2+Q_3+1}$. Through Forking lemma, the probability of $\mathcal{A}_2$ creating two valid forge signatures δ_1^* and δ_2^* is $\Pr[F_3] \geq \Big(1 - \frac{1}{e}\Big)\frac{\varepsilon_2}{Q'_h}$. Then, we have $\Pr[F_1 \wedge F_2 \wedge F_3] \geq \Big(1 - \frac{1}{e}\Big)\frac{\varepsilon_1}{e(Q_2+Q_3+1)Q'_h}$.

In conclusion, if there exists an adversary $\mathcal{A}_2$ who can break the unforgeability of our CLS scheme with advantage ε_2, then there exists a challenger $\mathcal{C}$ who can solve the hardness of DL assumption with non-negligible probability $(1-\frac{1}{e})\frac{\varepsilon_2}{e(Q_2+Q_3+1)Q'_h}$.

Theorem 3: If there exists a Type-III adversary $\mathcal{A}_3$ who can break the Non-repudiation of the CLS scheme with obvious advantage ε_3, then there exists a challenger $\mathcal{C}$ who can break the ElGamal signature scheme with non-negligible advantage $\frac{\varepsilon_3}{e(Q_1+Q_2+1)Q_h}$. We denote the queries times of H_1 oracle, partial private key generation and private key generation as Q_h, Q_1, Q_2, and e is the natural log base.

Proof 3: Let $\mathcal{A}_3$ be a Type-III adversary who can forge the public-private key pair with obvious advantage ε_3. Now, we construct a challenger $\mathcal{C}$ who can break the ElGamal signature scheme by using $\mathcal{A}_3$. The game between $\mathcal{C}$ and $\mathcal{A}_3$ is the following:

(1) Initialize. $\mathcal{C}$ randomly selects $a, h \in Z_q^*$ and sets $P_{pub} = sP'$ where P' is the public key of ElGamal signature scheme and $s \in Z_q^*$ is the master secret key. $\mathcal{C}$ initializes four lists L_2, L_3, L_4 and L_u used to record the queried message. Finally, $\mathcal{C}$ returns $Params = \{G, q, P, P_{pub}, H_0, H_1, H_2, H_3\}$ to $\mathcal{A}_1$, where H_1, H_2 and H_3 are three random oracles.

(2) Queries. $\mathcal{A}_3$ adaptively asks the queries as follows at this stage. The challenger $\mathcal{C}$ responds to H_2, H_3, and H_4 queries in the same way as Theorem 1. Furthermore, $\mathcal{C}$ chooses a random identity ID^* from the space submitted by $\mathcal{A}_3$ as the challenge identity.

Creation User. When $\mathcal{A}_3$ submits the query of creation user with identity ID as input, if $(ID, X, R, x, d) \in L_u$, $\mathcal{C}$ outputs the public key $PK = (X, R)$ to $\mathcal{A}_3$; Otherwise, $\mathcal{C}$ selects random values $r, x \in Z_q^*$, calculates $R = rP$, $X = xP$ and $d = r + sH_1(ID, X, R, P_{pub})$, then adds (ID, X, R, x, d) into the list L_u. Finally, $\mathcal{C}$ returns $PK = (X, R)$ to $\mathcal{A}_3$.

Partial Private Key. When $\mathcal{A}_3$ submits the partial private key generation query with inputting ID, $\mathcal{C}$ will abort if $ID = ID^*$; Otherwise, $\mathcal{C}$ uses the ID as index to seek a tuple (ID, X, R, x, d) from L_u , if it exists, $\mathcal{C}$ returns (d, R) to $\mathcal{A}_3$; else, $\mathcal{C}$ runs the creation user query for ID and outputs (d, R) to $\mathcal{A}_3$.

$Secret\ Value$. When $\mathcal{A}_3$ submits a secret value generation query with ID as input, if $ID = ID^*$, then $\mathcal{C}$ aborts; Otherwise, $\mathcal{C}$ seeks (ID, X, R, x, d) from L_u with ID as index, if it exists, then $\mathcal{C}$ returns x to $\mathcal{A}_3$; else, $\mathcal{C}$ will execute the query of creation user for ID and outputs x to $\mathcal{A}_3$.

$Private\ Key$. When $\mathcal{A}_3$ submits the query of private key generation with inputting ID, if $ID = ID^*$, $\mathcal{C}$ aborts; Otherwise, $\mathcal{C}$ make ID as index to seeks the tuple (ID, X, R, x, d) from L_u, if it exists, $\mathcal{C}$ returns $SK = (x, d)$ to $\mathcal{A}_3$; else, $\mathcal{C}$ executes the creation user query for ID and returns $SK = (x, d)$ to $\mathcal{A}_3$.

$Signature$. When $\mathcal{A}_3$ submits a signature generation query with inputting (ID, m_i), $\mathcal{C}$ performs the following operations.

1) If $ID \neq ID^*$, $\mathcal{C}$ chooses $t \in Z_q^*$, then calculates $T = tP$, $h_2 = H_2(P_{pub}, ID, m, PK)$, $h_3 = H_3(P_{pub}, ID, m, PK, T)$ and $h_4 = H_4(P_{pub}, ID, m, PK, T)$. Next $\mathcal{C}$ calculates $\sigma = h_4t + h_3d + h_2x$ and outputs $\delta = (T, \sigma)$ as signature to $\mathcal{A}_3$.

2) Otherwise, $\mathcal{C}$ selects $\sigma, h_2, h_3, h_4 \in Z_q^*$, and calculates $T = \frac{1}{h_4}\Big(\sigma P - h_3(R + h_1P_{pub}) - h_2X\Big)$, where $h_1 = H_1(ID, X, R, P_{pub})$. After that, adds $(ID, PK, P_{pub}, m_i, T, h_2)$, $(ID, PK, P_{pub}, m_i, T, h_3)$ and $(ID, PK, P_{pub}, m_i, T, h_4)$ into the lists L_2, L_3 and L_4. Finally, $\mathcal{C}$ outputs the signature $\delta = (T, \sigma)$ to $\mathcal{A}_3$.

(3) Forgery. Finally, $\mathcal{A}_3$ returns a forged public-private key pair (PK^*, SK^*) for the challenge identity ID_i, where $SK^* = (x^*, d^*) = (x^*, r^* + sH_1(ID, R^*, P_{pub}))$ and $PK^* = (X^*, R^*) = (x^*P, r^*P)$. If $ID_i \neq ID^*$, then $\mathcal{C}$ aborts; Otherwise, $\mathcal{C}$ outputs (R^*, d^*) as a forged signature of ElGamal scheme on the challenge message (ID^*, X^*, P_{pub}).

Notice that, if (PK^*, SK^*) is a valid public-private key pair, then (R^*, d^*) is a valid forged signature of the ElGamal scheme on the challenge message (ID^*, X^*, P_{pub}), because the equation $d^*P' = R^* + H_1(ID^*, X^*, R^*, P_{pub})P_{pub}$ holds.

We use $\mathcal{N}_1$ to indicate that at the query stage the above game does not abort; $\mathcal{N}_2$ indicates that at the forgery stage the above game does not abort; $\mathcal{N}_3$ indicates $\mathcal{A}_3$ creating valid key pair (PK^*, SK^*), that means (PK^*, SK^*) is the valid signature of ElGamal signature scheme for the message (ID^*, X^*, P_{pub}). Therefore, we get $\Pr[\mathcal{N}_1] = \Big(1 - \frac{1}{Q_1+Q_2+1}\Big)^{Q_1+Q_2}$, $\Pr[\mathcal{N}_2] = \frac{1}{Q_1+Q_2+1}$ and $\Pr[\mathcal{N}_3] = \varepsilon_3$. Then, we have $\Pr[\mathcal{N}_1 \wedge \mathcal{N}_2 \wedge \mathcal{N}_3] \geq \frac{\varepsilon_3}{e(Q_1+Q_2+1)Q_h}$.

In conclusion, if there exists an adversary $\mathcal{A}_3$ who can break the non-repudiation of our CLS scheme with advantage ε_3, then there exists a challenger $\mathcal{C}$ who can break the ElGamal signature scheme with non-negligible probability $\frac{\varepsilon_3}{e(Q_1+Q_2+1)Q_h}$.

D. Performance Analysis of Our Proposal

In this section, we will compare our CLS scheme with other CLS schemes integrally from the aspects of computation and communication. We use the Charm-crypto library to perform the simulation experiment based on pycharm, where our algorithm runs on Linux, the processor is Intel i7-8550, and the memory is 8G. We show the the base operations' cost in Table II.

TABLE II
RUNNING COST OF BASE OPERATIONS

Symbol	Algorithm	Time Required (ms)
T_{pa}	A point addition on elliptic curve	0.0021
T_{ps}	A point subtraction on elliptic curve	0.0021
T_{sm}	A scalar multiplication on elliptic curve	0.4412
T_{me}	A modular exponentiation operation	3.7043
T_{rc}	A modular inversion computation	0.1920
T_{bp}	Bilinear Pairing operation	2.5521

1) Communication Cost Comparison: The signature length is a prominent element in deciding the communication cost. We use $|G|$ to express the size of an additional group $|G|$, $|G_1|$ to denote the size of a multiplicative group $|G_1|$, $|Z_q^*|$ to express the group element's size in the finite field. Hence, our communication cost is $|G| + |Z_q^*|$ (480 bits). But through our previous security analysis, the corresponding constructions [15], [16], [21], [27] are vulnerable to the Type-I adversaries' attack, and scheme [21] can not resist the Type-II adversary. Furthermore, the length of signature [18], [26], [28] is too long, so they have high communication costs. Therefore, our scheme achieves lower communication costs and better security. We use Table III to describe the comparison results of communication costs.

TABLE III
COMPARISON OF COMMUNICATION COST

Scheme	Signature length	Communication cost
Scheme [15]	$\lvert G\rvert$+ $\lvert Z_q^*\rvert$	480bits
Scheme [16]	$\lvert G\rvert$+ $\lvert Z_q^*\rvert$	480bits
Scheme [18]	$\lvert G_1\rvert$+ $\lvert Z_q^*\rvert$	1184bits
Scheme [19]	$\lvert G\rvert$+ $\lvert Z_q^*\rvert$	480bits
Scheme [21]	$\lvert G\rvert$+ $\lvert Z_q^*\rvert$	480bits
Scheme [26]	$2\lvert G_1\rvert$	2048bits
Scheme [27]	$\lvert G\rvert$+ $\lvert Z_q^*\rvert$	480bits
Scheme [28]	$2\lvert G_1\rvert$	2048bits
our scheme	$\lvert G\rvert$+ $\lvert Z_q^*\rvert$	480bits

2) Computation Cost Comparison: In this part, we mainly consider the cost of the signature and verification algorithms. From table II, table IV, and Fig. 2, we can find that the computation cost of our scheme is $5T_{sm} + 2T_{pa} + T_{ps} = 2.0623(ms)$. Based on Table II, we can obtain the proposals [18], [26], [28] contain bilinear mapping operation, and they have the highest calculation costs. Although the computation efficiency of constructions is low, they can not resist the signature forgery attack [15], [16], [19], [21], [27]. Furthermore, the existing constructs do not have anonymity [15]–[17], [19].

Therefore, our CLS scheme has high security and computation efficiency and can realize anonymity. Therefore, our scheme is the most suitable for health IIoT with limited resources through comprehensive comparison.

V. A Secure Identity Authentication Protocol from Our CLS Scheme

In healthcare IIoT, patients and doctors can achieve timely treatment by sharing health data and diagnosis programs.

TABLE IV
COMPREHENSIVE COMPARISION

Scheme	Signature Generation	Verification	Security
scheme [15]	$1T_{sm}$+$1T_{rc}$	$4T_{sm}$+$2T_{pa}$	no
scheme [16]	$1T_{sm}$+$1T_{rc}$	$4T_{sm}$+$2T_{pa}$	no
scheme [18]	$1T_{me}$	$4T_{me}$	yes
scheme [19]	$1T_{sm}$	$4T_{sm}$+$2T_{pa}$	yes
scheme [21]	$2T_{sm}$	$3T_{sm}$+$1T_{pa}$+$1T_{ps}$	no
scheme [26]	$2T_{sm}$	T_{bp}+T_{me}	yes
scheme [27]	$1T_{sm}$	$4T_{sm}$+T_{pa}	no
scheme [28]	$2T_{sm}$	T_{bp}+T_{me}	yes
our scheme	$1T_{sm}$	$4T_{sm}$+$1T_{pa}$+$1T_{ps}$	yes

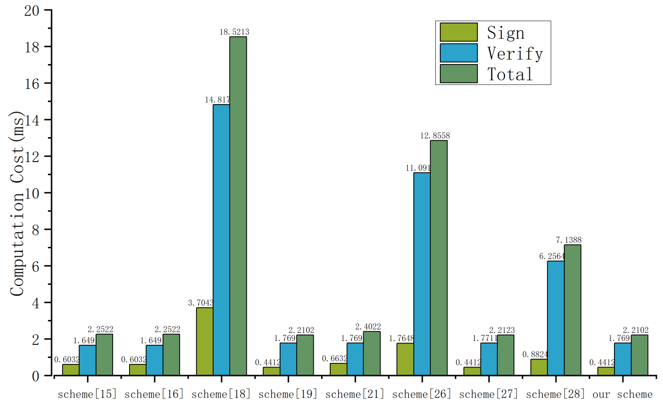


Fig. 2. The comparison results of computation cost

However, to ensure the identity legality of communication entities, patients and doctors should verify each other's identities. Therefore, we propose an identity authentication and key agreement protocol utilizing our CLS scheme that can realize mutual identity authentication between patients and doctors, which can ensure identity legality and data confidentiality. Furthermore, we prove its security and effectiveness through security simulation and performance analysis.

A. Concrete Construction

Initialization. KGC selects an additive group G of order q and the generator is P, and selects $s \in Z_q^*$ to compute $P_{pub} = sP$. Let $H, H_1, H_2, H_3, H_4, H_5 : \{0,1\}^* \rightarrow Z_q^*$ be five one-way hash functions. Finally, KGC sets $Params$=$\{G, q, p, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$ as the public parameter.

Registration. The registration operations of the doctor and patient are as follows:

(1) Doctor or patient sends the corresponding identity RID to KGC, after receiving the ID, KGC computes the pseudonym $ID = RID \oplus H_0(rP_{pub}, V_t)$, where V_t is the validity period of ID. Then, KGC selects $r \in Z_q^*$ to compute $R = rP$ and $d = r + sh_1 \mod q$, where $h_1 = H_1(P_{pub}, ID, R)$, then KGC sets the (d, R) as the partial key.

(2) The doctor or patient can verify the partial key's correctness by calculating $dP = R + h_1 P_{pub}$ and determining the validity of pseudonym ID. Afterwards, doctor or patient sets $x \in Z_q^*$ as the secret value, computes $X = xP$ and sets the private key $SK = (x, d)$ and the public key $PK = (X, R)$.

Identity authentication. In this part, the doctor and patient achieve mutual identity verification. Define U_a and U_b denote the patient and doctor, Enc and Dec are the algorithms of the symmetric key encryption and symmetric key decryption.

(1) The authentication request operations of U_a are as follows. Let $Time_a$ be the current time stamp. Select $t_a, z_a \in Z_q^*$ to calculate $T_a = t_a P$, $Z_a = z_a P$, $W_a = t_a(X_b + R_b + h_1^b P_{pub})$ and $C_a = \mathsf{Enc}(k_a, T_a || ID_a || Time_a)$, where $h_1^b = H_1(P_{pub}, ID_b, R_b)$ and $k_a = H(W_a)$. Afterward, compute $\sigma_a = h_4^a t_a + h_3^a d_a + h_2^a x_a \mod q$, where $h_2^a = H_2(P_{pub}, ID_a, PK_a, T_a, C_a)$, $h_3^a = H_3(P_{pub}, ID_a, PK_a, T_a, C_a)$ and $h_4^a = H_4(P_{pub}, ID_a, PK_a, T_a, C_a)$. Finally, U_a sends $(T_a, Z_a, C_a, \sigma_a, Time_a)$ to cloud server.

(2) The verification operations of U_b are as follows. Download $(T_a, Z_a, C_a, \sigma_a, Time_a)$ from cloud server. Compute $W_a' = (x_b + d_b)T_a$ and $T_a || ID_a || Time_a = \mathsf{Dec}(k_a', W_a')$, where $k_a' = H(W_a')$, where $Time_a$ is employed to avoid the replay attacks. Get the decryption algorithm through computing $W_a' = (x_b + d_b)T_a = t_a(x_b + d_b)P = t_a(X_b + R_b + h_1^b P_{pub}) = W_a$. If $\sigma_a P = h_4^a T_a + h_3^a(R_a + h_1^a P_{pub}) + h_2^a X_a$, then the validity of U_a is verified.

(3) The authentication request operations of U_b are the following. Let $Time_b$ be the current time stamp. select $t_b, z_b \in Z_q^*$ to calculate $T_b = t_b P$, $Z_b = z_b P$, $W_b = t_b(X_a + R_a + h_1^a Ppub)$ and $C_b = \mathsf{Enc}(k_b, T_b || ID_b, Time_b)$, where $h_1^a = H_1(P_{pub}, id_a, R_a)$ and $k_b = H(W_b)$. Afterward, compute $\sigma_b = h_4^b t_b + h_3^b d_b + h^a b_2 x_b$ and q, where $h_2^b = H_2(P_{pub}, id_b, PK_b, T_b, C_b)$, $h_3^b = H_3(P_{pub}, id_b, PK_b, T_b, C_b)$ and $h_4^b = H_4(P_{pub}, id_b, PK_b, T_b, C_b)$. Finally, U_b sends $(T_b, Z_b, C_b, \sigma_b, Time_b)$ to cloud server.

(4) The verification operations of U_a are as follows. Download $(T_b, Z_b, C_b, \sigma_b, Time_b)$ from cloud server. Compute $W_b' = (x_a + d_a)T_b$ and $T_b || ID_b || Time_b = \mathsf{Dec}(k_b', W_b')$, where $k_b' = H(W_b')$, where $Time_b$ is also employed to avoid the replay attacks.. Get the decryption algorithm through computing $W_b' = (x_a + d_a)T_b = t_b(x_a + d_a)P = t_b(X_a + R_a + h_1^a P_{pub}) = W_b$. If $\sigma_b P = h_4^b T_b + h_3^b(R_b + h_1^b P_{pub}) + h_2^b X_b$ holds, then the validity of U_b is verified.

Key Agreement After verifying the legitimacy of mutual identity, U_a and U_b make the session key K_{ab}.

(1) U_a calculates the $K_a = Z_b z_a = z_b z_a P$, then computes the session key $K_{ab} = H_5(U_a, U_b, K_a)$

(2) U_b calculates the $K_b = Z_a z_b = z_a z_b P$, then computes the session key $K_{ab} = H_5(U_a, U_b, K_b)$

Finally, U_a and U_b utilize the session key K_{ab} to encrypt and decrypt messages.

B. Security Analysis

(1) Mutual authentication. Only legitimate U_a and U_b can generate legitimate request messages. Our CLS scheme can ensure the validity of the message. In the previous introduction, we also proved that our scheme is unforgivable. Therefore, the adversary cannot forge the communication messages of the identity authentication protocol.

(2) Resist replay attacks. In the identity authentication protocol, we use the time stamps $Time_a$ and $Time_b$ used in our protocol to resist replay attacks.

TABLE V
COMPARISON OF PROTOCOL COMPUTATION COST

Scheme	Registration	Authentication	Total
scheme [31]	$3T_{pbsm}$	$5T_{pbsm}$+T_{pba}+T_{bp}	$8T_{pbsm}$+T_{pba}+T_{bp}
scheme [32]	$4T_{pbsm}$	$2T_{pbsm}$+$3T_{bp}$+$2T_{me}$	$6T_{pbsm}$+$3T_{bp}$+$2T_{me}$
scheme [33]	$2T_{pbsm}$+$3T_{bp}$+T_{me}	T_{pbsm}+T_{bp}+$2T_{me}$	$3T_{pbsm}$+T_{bp}+$3T_{me}$
scheme [34]	$7T_{pbsm}$	$2T_{bp}$	$7T_{pbsm}$+$2T_{bp}$
our scheme	$5T_{sm}$+T_{pa}	$5T_{sm}$+$4T_{pa}$	$10T_{sm}$+$5T_{pa}$

(3) Forward and backward security. Because the value of a random number for creating the session key in each request and response process is different, even if a session key leaks, it will not affect the security of the other. Therefore, our protocol has excellent forward and backward security.

(4) Unlinkability. When U_a sends a message to U_b, in our scheme, W_a and T_a are two random numbers. Because each request has different values, the adversary does not know the message is sent by the same U_a, so our scheme has unlinkability.

(5) Anonymity. To ensure the privacy of doctors and patients that protect their identities do not leak, we use pseudonyms instead of their true identities and set a validity period for them. Once the timestamp expires, the pseudonyms will be invalidated. So, our protocol realizes anonymity.

(6) Confidentiality. After verifying the legitimacy of both parties' identities, the participant generates a session key. In subsequent communication, the user employs the above session key to encrypt the message, ensuring the confidentiality of the data.

(7) Integrity. The above protocol is created based on our CLS scheme, which ensures that the data has not been modified and protects the integrity of the data through a certificateless signature.

(8) Resist MITM attack. Our protocol employs the underlying CLS scheme to prevent man-in-the-middle attacks, which can ensure the messages are unmodified. Furthermore, we use encrypted messages through session keys to prevent message leakage. Therefore, the malicious attackers can not intercept and modify the communication messages.

C. Performance Analysis

In this section, we compare the calculation cost of our identity authentication protocol with several identity authentication protocols. We found that Jiang *et al.*'s protocol [32] is vulnerable to replay attacks. Rasslan *et al.* [33] designed an identity-based protocol for the medical Internet of things, which existed as a key-escrow problem. We also found that our mutual identity authentication protocol using a pairing-free CLS scheme has better efficiency from Fig.3 and Table V, while the calculation cost of the existing protocols [31], [34], based on bilinear mapping is too high.

D. Simulation Analysis

We conducted simulation experiments to apply our scheme to identity authentication and key exchange protocols based on medical systems, in which, if the patient and doctor log in, they send their identities to each other, which can be verified based on fuzzy queries, and they generate a session key to encrypt the transmitted message.

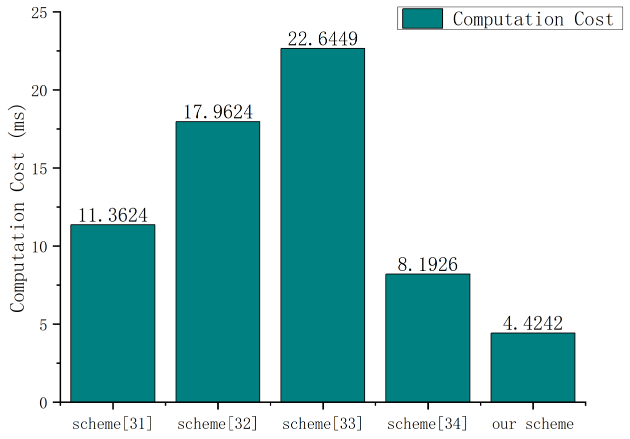


Fig. 3. The comparison results of computation cost of identity authentication

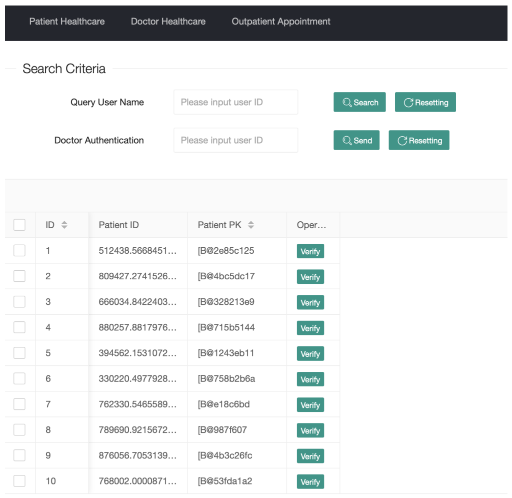
We conducted simulation experiments to apply our protocol to achieve identity authentication and key exchange based on medical systems. After the patient and doctor log in, they send their identities to each other, which can be verified based on fuzzy queries. Finally, we generate a session key used to encrypt the transmitted message. The above simulation experiment runs on a Windows operation system, with the backend using the SSM framework (Spring+SpringMVC+MyBatis) and the front end using the View framework and Element components, and we use the MySql database to store user identity information. For further convenience to the reader, we have uploaded the specific running code to Github [1]. The following is the screenshot of our system operation.

Fig. 4 is the user interface, where patients and doctors enter interfaces through their respective accounts. The patient can query the identity sent by the doctor and send their ID to the doctor for verification, and we use Fig. 5 to describe the above operations. The doctor conducts a fuzzy query to find the ID sent by the patient, and we use Fig. 6 to describe the above operations. Similarly, the doctor can send the ID to the patient to perform identity authentication, and we use Fig. 7 to show the verification process. Querying the user's public key through the user ID shows the process of doctors verifying the patient's identity, and the authentication method of patients verifying the doctor's identity is the same. Fig. 8 shows verification successes, user identity authentication has passed, and after both parties' identity authentication passes, the participant securely transmits the communication message.

VI. Conclusions

The security advantage of CLS makes it widely used in health IIoT. In this paper, we prove that the existing CLS scheme created by Wang *et al.* [21] is vulnerable to the signature attack of the Type-II adversary. We propose a novel

[1]https://github.com/XRstar1/Healthcare-System.git

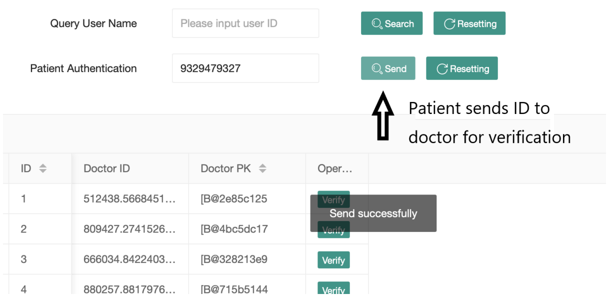


Query User Name: Please input user ID

Doctor Authentication: Please input user ID

	ID	Patient ID	Patient PK	Oper...
	1	512438.5668451...	[B@2e85c125	Verify
	2	809427.2741526...	[B@4bc5dc17	Verify
	3	666034.8422403...	[B@328213e9	Verify
	4	880257.8817976...	[B@715b5144	Verify
	5	394562.1531072...	[B@1243eb11	Verify
	6	330220.4977928...	[B@758b2b6a	Verify
	7	762330.5465589...	[B@e18c6bd	Verify
	8	789690.9215672...	[B@987f607	Verify
	9	876056.7053139...	[B@4b3c26fc	Verify
	10	768002.0000871...	[B@53fda1a2	Verify

Fig. 4. User interface of our system

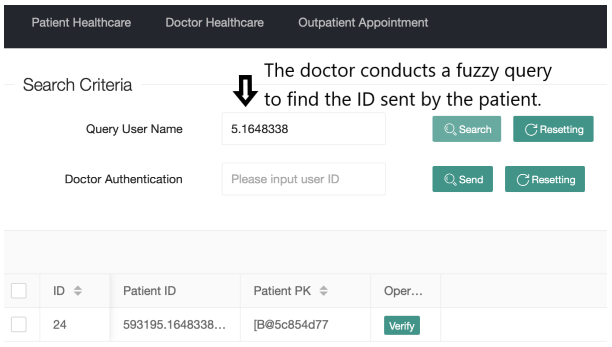


Query User Name: Please input user ID

Patient Authentication: 9329479327

ID	Doctor ID	Doctor PK	Oper...
1	512438.5668451...	[B@2e85c125	Verify
2	809427.2741526...	[B@4bc5dc17	Verify
3	666034.8422403...	[B@328213e9	Verify
4	880257.8817976...	[B@715b5144	Verify

Fig. 5. Patients send ID to doctors for verification

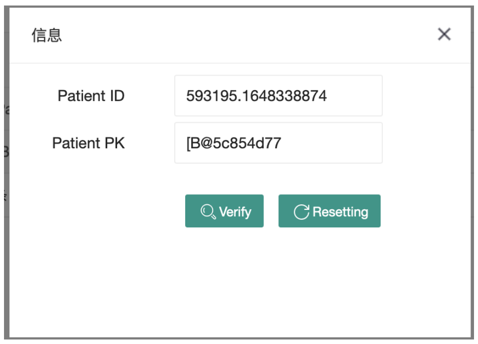


Query User Name: 5.1648338

Doctor Authentication: Please input user ID

	ID	Patient ID	Patient PK	Oper...
	24	593195.1648338...	[B@5c854d77	Verify

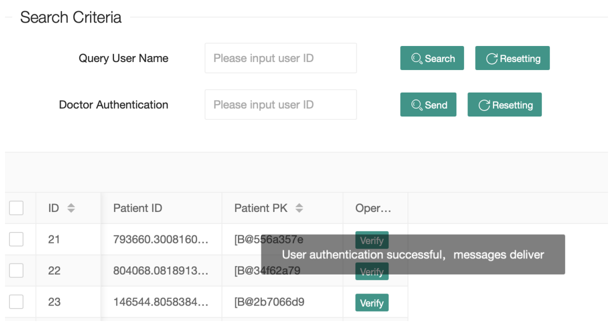
Fig. 6. Doctors inquire about ID from patients



Patient ID: 593195.1648338874

Patient PK: [B@5c854d77

Fig. 7. The doctor verifies the legitimacy of the patient's identity



Query User Name: Please input user ID

Doctor Authentication: Please input user ID

	ID	Patient ID	Patient PK	Oper...
	21	793660.3008160...	[B@556a357e	Verify
	22	804068.0818913...	[B@34f62a79	Verify
	23	146544.8058384...	[B@2b7066d9	Verify

Fig. 8. The patient's identity is verified successfully

CLS scheme and prove it has achieved Giraults level 3 security. The performance analysis shows that our proposal has less computation and communication costs. Hence, we believe that our scheme is suitable for resource-constrained environments. In addition, we utilize our CLS scheme to build a mutual identity authentication and key agreement protocol to confirm the legitimacy of user identity and protect the security of messages in real applications. To sum up, our CLS scheme applies to the application of IIoT technology in interconnected medical devices and sensors.

REFERENCES

[1] Praveen Kumar Malik, Rohit Sharma, Rajesh Singh, Anita Gehlot, Suresh Chandra Satapathy, Waleed S Alnumay, Danilo Pelusi, Uttam Ghosh, and Janmenjoy Nayak. Industrial internet of things and its applications in industry 4.0: State of the art. *Computer Communications*, 166:125–139, 2021.

[2] Wazir Zada Khan, MH Rehman, Hussein Mohammed Zangoti, Muhammad Khalil Afzal, Nasrullah Armi, and Khaled Salah. Industrial internet of things: Recent advances, enabling technologies and open challenges. *Computers & Electrical Engineering*, 81:106522, 2020.

[3] Martin Serror, Sacha Hack, Martin Henze, Marko Schuba, and Klaus Wehrle. Challenges and opportunities in securing the industrial internet of things. *IEEE Transactions on Industrial Informatics*, 17(5):2985–2996, 2020.

[4] Giuseppe Aceto, Valerio Persico, and Antonio Pescapé. Industry 4.0 and health: Internet of things, big data, and cloud computing for healthcare 4.0. *Journal of Industrial Information Integration*, 18:100129, 2020.

[5] Damini Verma, Kshitij RB Singh, Amit K Yadav, Vanya Nayak, Jay Singh, Pratima R Solanki, and Ravindra Pratap Singh. Internet of things (iot) in nano-integrated wearable biosensor devices for healthcare applications. *Biosensors and Bioelectronics: X*, 11:100153, 2022.

[6] Mina Younan, Essam H Houssein, Mohamed Elhoseny, and Abdelmgeid A Ali. Challenges and recommended technologies for the industrial internet of things: A comprehensive review. *Measurement*, 151:107198, 2020.

[7] Sattam S Al-Riyami, Kenneth G Paterson, et al. Certificateless public key cryptography. In *Asiacrypt*, volume 2894, pages 452–473. Springer, 2003.

[8] Marc Girault. Self-certified public keys. In *Advances in CryptologyEUROCRYPT91: Workshop on the Theory and Application of Cryptographic Techniques Brighton, UK, April 8–11, 1991 Proceedings 10*, pages 490–497. Springer, 1991.

[9] Yu-Chi Chen, Raylin Tso, Gwoboa Horng, Chun-I Fan, Ruei-Hau Hsu, et al. Strongly secure certificateless signature: Cryptanalysis and improvement of two schemes. *J. Inf. Sci. Eng.*, 31(1):297–314, 2015.

[10] Arijit Karati, Chun-I Fan, and Er-Shuo Zhuang. Reliable data sharing by certificateless encryption supporting keyword search against vulnerable kgc in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 18(6):3661–3669, 2021.

[11] Saddam Hussain, Syed Sajid Ullah, Ihsan Ali, Jiafeng Xie, and Venkata N Inukollu. Certificateless signature schemes in industrial internet of things: A comparative survey. *Computer Communications*, 181:116–131, 2022.

[12] Yuan Xu, Yanwei Zhou, Bo Yang, Zirui Qiao, Zhaolong Wang, Zhe Xia, and Mingwu Zhang. An efficient identity authentication scheme with provable security and anonymity for mobile edge computing. *IEEE System Journal*, 17(1):1012–1023, 2023.

[13] Ikram Ali, Yong Chen, Mohammad Faisal, Meng Li, Ikram Ali, Yong Chen, Mohammad Faisal, and Meng Li. Certificateless signature-based authentication scheme for vehicle-to-infrastructure communications using bilinear pairing. *Efficient and Provably Secure Schemes for Vehicular Ad-Hoc Networks*, pages 91–119, 2022.

[14] Xiaoying Jia, Debiao He, Qin Liu, and Kim-Kwang Raymond Choo. An efficient provably-secure certificateless signature scheme for internet-of-things deployment. *Ad Hoc Networks*, 71:78–87, 2018.

[15] Hongzhen Du, Qiaoyan Wen, Shanshan Zhang, and Mingchu Gao. A new provably secure certificateless signature scheme for internet of things. *Ad Hoc Networks*, 100:102074, 2020.

[16] Dengmei Xiang, Xuelian Li, Juntao Gao, and Xiachuan Zhang. A secure and efficient certificateless signature scheme for internet of things. *Ad Hoc Networks*, 124:102702, 2022.

[17] Kui Ma, Yanwei Zhou, Ying Wang, Chunsheng Dong, Zhe Xia, Bo Yang, and Mingwu Zhang. An efficient certificateless signature scheme with provably security and its applications. *IEEE Systems Journal*, 2023.

[18] Liangliang Wang, Kefei Chen, Yu Long, and Huige Wang. An efficient pairing-free certificateless signature scheme for resource-limited systems. *Science China Information Sciences*, 60(11):1–3, 2017.

[19] Gowri Thumbur, G Srinivasa Rao, P Vasudeva Reddy, NB Gayathri, and DV Rama Koti Reddy. Efficient pairing-free certificateless signature scheme for secure communication in resource-constrained devices. *IEEE Communications Letters*, 24(8):1641–1645, 2020.

[20] Zhiyan Xu, Min Luo, Muhammad Khurram Khan, Kim-Kwang Raymond Choo, and Debiao He. Analysis and improvement of a certificateless signature scheme for resource-constrained scenarios. *IEEE Communications Letters*, 25(4):1074–1078, 2020.

[21] Weizheng Wang, Hao Xu, Mamoun Alazab, Thippa Reddy Gadekallu, Zhaoyang Han, and Chunhua Su. Blockchain-based reliable and efficient certificateless signature for iiot devices. *IEEE transactions on industrial informatics*, 18(10):7059–7067, 2021.

[22] Shuanggen Liu, Zhifeng Wan, Yixin Yuan, Qiyue Dong, Baozhu Yang, and Fei Hao. An efficient certificateless blind signature scheme with conditional revocation for mobile crowd sensing within smart city. *IEEE Internet of Things Journal*, 2024.

[23] Kyung-Ah Shim. A secure certificateless signature scheme for cloud-assisted industrial iot. *IEEE Transactions on Industrial Informatics*, 2024.

[24] Kyung-Ah Shim. Design principles of secure certificateless signature and aggregate signature schemes for iot environments. *IEEE Access*, 10:124848–124857, 2022.

[25] Wenjie Yang, Shangpeng Wang, and Yi Mu. An enhanced certificateless aggregate signature without pairings for e-healthcare system. *IEEE Internet of Things Journal*, 8(6):5000–5008, 2020.

[26] Yangfan Liang and Yining Liu. Analysis and improvement of an efficient certificateless aggregate signature with conditional privacy preservation in vanets. *IEEE Systems Journal*, 17(1):664–672, 2023.

[27] Wanjun Xiong, Ruomei Wang, Yujue Wang, Yongzhuang Wei, Fan Zhou, and Xiaonan Luo. Improved certificateless aggregate signature scheme against collusion attacks for vanets. *IEEE Systems Journal*, 17(1):1098–1109, 2023.

[28] Vijay Kumar Yadav, Nitish Andola, Shekhar Verma, and S. Venkatesan. PSCLS: provably secure certificateless signature scheme for iot device on cloud. *The Journal of Supercomputing*, 79(5):4962–4982, 2023.

[29] Bessie C Hu, Duncan S Wong, Zhenfeng Zhang, and Xiaotie Deng. Key replacement attack against a generic construction of certificateless signature. In *Australasian Conference on Information Security and Privacy*, pages 235–246. Springer, 2006.

[30] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of cryptology*, 13:361–396, 2000.

[31] Tsu-Yang Wu, Yu-Qi Lee, Chien-Ming Chen, Yuan Tian, and Najla Abdulrahman Al-Nabhan. An enhanced pairing-based authentication scheme for smart grid communications. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–13, 2021.

[32] Yili Jiang, Kuan Zhang, Yi Qian, and Liang Zhou. Anonymous and efficient authentication scheme for privacy-preserving distributed learning. *IEEE Transactions on Information Forensics and Security*, 17:2227–2240, 2022.

[33] Mohamed Rasslan, Mahmoud M Nasreldin, and Heba K Aslan. Ibn sina: A patient privacy-preserving authentication protocol in medical internet of things. *Computers & Security*, 119:102753, 2022.

[34] Irfan Alam and Manoj Kumar. A novel authentication protocol to ensure confidentiality among the internet of medical things in covid-19 and future pandemic scenario. *Internet of Things*, 22:100797, 2023.