sensors

MDPI

Article

A Strongly Unforgeable Certificateless Signature Scheme and Its Application in IoT Environments

Xiaodong Yang *, Xizhen Pei, Guilan Chen, Ting Li, Meiding Wang and Caifen Wang

Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China; pxz0626@163.com (X.P.); chenglpaper@163.com (G.C.); lt1358382@163.com (T.L.); wmding95@163.com (M.W.); wangcfen@126.com (C.W.)

* Correspondence: yangxd@nwnu.edu.cn

Received: 4 April 2019; Accepted: 11 June 2019; Published: 14 June 2019

Abstract: With the widespread application of the Internet of Things (IoT), ensuring communication security for IoT devices is of considerable importance. Since IoT data are vulnerable to eavesdropping, tampering, forgery, and other attacks during an open network transmission, the integrity and authenticity of data are fundamental security requirements in the IoT. A certificateless signature (CLS) is a viable solution for providing data integrity, data authenticity, and identity identification in resource-constrained IoT devices. Therefore, designing a secure and efficient CLS scheme for IoT environments has become one of the main objectives of IoT security research. However, the existing CLS schemes rarely focus on strong unforgeability and replay attacks. Herein, we design a novel CLS scheme to protect the integrity and authenticity of IoT data. In addition to satisfying the strong unforgeability requirement, the proposed scheme also resists public key replacement attacks, malicious-but-passive key-generation-centre attacks, and replay attacks. Compared with other related CLS schemes without random oracles, our CLS scheme has a shorter private key, stronger security, and lower communication and computational costs.

Keywords: certificateless signature; the Internet of Things; data integrity; data authenticity; strong unforgeability; provable security

1. Introduction

The Internet of Things (IoT) is a self-establishing network of smart devices that are equipped with electronics, sensors, software, and actuators and that are connected via the Internet to generate, collect, and exchange data [1]. Since IoT devices connect objects in different environments to the Internet for information exchange and communication to realize intelligent identification, location, tracking, monitoring, management, and other functions, IoT devices have the ability to support a wide range of services. Consequently, the IoT builds a network that covers various things throughout the world via numerous IoT devices, and it enables various human-to-human, human-to-thing, thing-to-thing, and thing-to-thing interactions. Figure 1 shows a variety of IoT applications, including intelligent transportation, military target tracking, surveillance, public safety, smart home, industrial monitoring, smart city, medical equipment, and food traceability [2]. The application of the IoT involves all economic and social aspects of daily life and fundamentally changes the way in which humans interact with the world around them. Hence, the IoT is considered to be an information technology revolution and has become a growth point for the global economy [3].

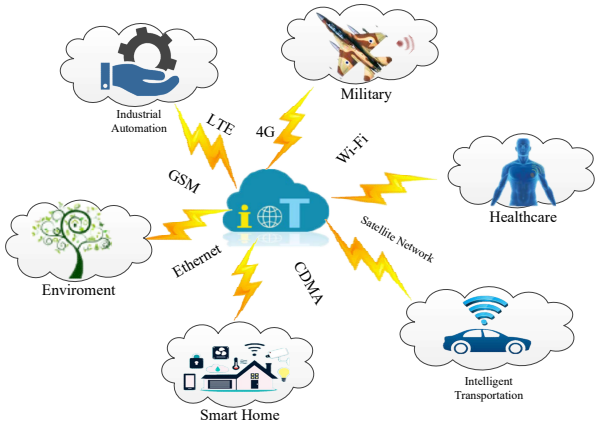

Figure 1. Internet of Things (IoT) applications.

Various IoT-enabled devices with embedded sensors collect and send IoT data to data centres over public networks; thus, the issues of security and privacy in IoT environments have become increasingly important [4,5]. Only authentic data can be stored in data centres, which requires the integrity and authenticity of the data transmitted by an IoT device to be checked before being stored. The signature-based cryptosystem is technology that provides the integrity, real source, unforgeability, and non-repudiation of the data. An IoT device signs the data using its private key during data transmission, and the data centre confirms the data authenticity and integrity by verifying the validity of the received signature. Therefore, a digital signature scheme can ensure data integrity and data authenticity in the IoT. However, the IoT differs from traditional networks. Most IoT devices have limited computational and processing capabilities, short communication ranges, and restricted storage and power resources. Conventional cryptosystems cannot run on resource-constrained IoT devices. The main reason is that conventional cryptosystems are classified into two categories: PKI-based and ID-based cryptosystems. Traditional PKI-based cryptosystems require certificates to authenticate users' public keys, which results in a large amount of computational overhead and communication costs to manage and exchange certificates. The identity-based cryptosystem avoids the use of certificates, but there are security flaws in key escrow that make it unsuitable for large-scale network environments. Hence, designing an efficient, secure signature scheme is very important for IoT security.

In a signature scheme, the private key of the signer is used to sign the message, and the validity of the corresponding signature is verified by the signer's public key. The signature's validity not only ensures that the signer with the private key can apply a valid signature to the message but also ensures the authenticity and integrity of the message. The user's public key is generally a random string; thus, authenticating the authenticity of the user's public key is particularly crucial. In traditional public key infrastructure (PKI) settings, a certificate issued by a fully trusted authority associates the user's public key with the user's real identity. The authenticity of the user's public key can be verified by the legality of the corresponding certificate. However, the storage, distribution, verification, and revocation of certificates in PKI are resource-intensive and computationally expensive tasks [6]. Hence, PKI is unsuitable for resource-constrained IoT environments.

Shamir [7] proposed identity-based cryptography (IBC) to solve the complex certificate management problems in PKI. IBC allows a key generation centre (KGC) to produce the user's private key, but the corresponding user's public key comes from their public identity information, such as an e-mail address or mobile phone number. However, KGC can replace the user to decrypt any ciphertext or to forge the signature of any message without being found, which results in the key escrow problem.

The concept of certificateless signature (CLS) was introduced by Al-Riyami and Paterson [8]. In a CLS scheme, the user's private key consists of two parts: One is a partial private key generated by the KGC, and the other is a secret value calculated independently by the user. The CLS scheme solves the key escrow problem because the KGC is unable to obtain the user's final private key. In addition, the user generates the corresponding public key based on its secret value, but it is not necessary to verify the authenticity of the public key by using the certificate. In practical applications, the user's public key is sent to the recipient together with the signature or is obtained from a public directory in a proper manner.

Certificateless signatures have received considerable attention in recent years, and researchers have designed numerous CLS schemes [9–12]. Most existing CLS schemes [13–15] have been proven to be secure in the random oracle model [16], where a cryptographic hash function is modelled as an ideal random oracle. The random oracle paradigm helps construct efficient cryptographic schemes, but it has received substantial criticism. It has been shown that, when random oracles are instantiated with actual hash functions, the cryptographic scheme that proves to be secure using the random oracle model may be unsafe in reality [17]. To overcome this security flaw, Liu et al. [18] designed the first CLS scheme without random oracles. Later, several CLS schemes [19–22] in the standard model were proposed, but these schemes cannot resist public key replacement (PKR) attacks or malicious-but-passive KGC (MKGC) attacks. In addition, most existing CLS schemes [23–25] without random oracles are proven to be existentially unforgeable against adaptive chosen-message attacks. This security notion only ensures that an attacker cannot forge the signature of any new message; it does not guarantee that the attacker generates the valid signature for the signed message. However, some signature schemes are malleable [26]; thus, an attacker can generate multiple valid signatures of the same message by using the previous message–signature pair without the signer's private key. In other words, these schemes do not satisfy strong unforgeability, which is a stronger security notion than existential unforgeability. Strong unforgeability is desirable in some applications [27–29] (such as electronic commerce, construction of certificateless signcryption schemes, and certificateless group signature schemes). If a CLS signature scheme satisfies existential unforgeability and can prevent an attacker from forging a valid signature of a previously signed message, then we say that the CLS scheme is strongly unforgeable. Strong unforgeability is an important property of the CLS scheme, but few CLS schemes [30] satisfy strong unforgeability in the standard model. Unfortunately, none of those strongly unforgeable CLS schemes considers replay attacks [31,32]. Note that the energy of the IoT device is one of the main factors that restricts improvements in network performance. However, replay attacks, which are considered to be one of the major attacks faced by IoT devices, can consume a large amount of node energy. Therefore, a CLS scheme that is applicable to IoT environments must consider replay attacks.

In this paper, motivated by the above concerns, we present a new CLS scheme for IoT environments that is more secure and efficient than the previous CLS schemes. As a potential signature-based authentication technology, our proposed scheme manifests a solution to the problems of data authenticity and data integrity in the IoT. The main contributions of this paper are the following.

- A novel CLS scheme without random oracles is constructed. Under the collision-resistant hash function (CRHF) and computational Diffie–Hellman (CDH) assumptions, the proposed CLS scheme is proven to be strongly unforgeable against adaptive chosen-message attacks in the standard model.
- In our CLS scheme, the user's public key is not only bound to the user's partial private key but also embedded into the signature of the message. This makes the proposed CLS scheme have a higher security trust level and be capable of resisting PKR attacks and MKGC attacks.
- The proposed CLS scheme resists replay attacks by verifying the freshness of the timestamp and the validity of the signature. To our best knowledge, our scheme is the first CLS scheme with a strong unforgeability in the standard model that can resist replay attacks.

- Compared to other CLS schemes in the standard model, our CLS scheme has higher security, a smaller key size, a shorter signature length, and lower computational overhead for signature generation and signature verification.
- Due to the aforementioned functionalities, our CLS scheme is able to be implemented and deployed in IoT environments where IoT devices have limited computing power, storage space, and communication bandwidth.

The remainder of this paper is organized as follows. We present the relevant CLS works in Section 2. Then, we introduce some preliminaries and security notions of the CLS scheme in Section 3. The proposed CLS scheme and its security proof are presented in Sections 4 and 5. Section 6 gives the CLS system model for IoT environments and performance analysis. Section 7 concludes this paper.

2. Related Work

The first CLS scheme was proposed by Al-Riyami and Paterson [8]. Later, Huang et al. [33] noted that their CLS scheme [8] was unable to resist PKR attacks and proposed security notions for CLS schemes. Since then, researchers have constructed a large number of provably secure CLS schemes [9–15] in the random oracle model. Aiming to eliminate the security requirements of ideal random oracles, Liu et al. [18] constructed a CLS scheme without random oracles based on the identity-based signature scheme proposed by Paterson and Schuldt [34]. However, Xiong et al. [19] and Huang et al. [35] demonstrated that Liu et al.'s CLS scheme [18] was insecure against MKGC attacks. To enhance the security of Liu et al.'s CLS scheme [18], Xiong et al. [19] presented an improved scheme, but it was still vulnerable to MKGC attacks [36]. Furthermore, Xia et al. [37] showed that several CLS schemes [18–20] without random oracles were susceptible to PKR attacks.

Subsequently, Yu et al. [21] designed a CLS scheme and claimed that their scheme was secure in the standard model. However, Yuan et al. [23] and Pang et al. [27] independently demonstrated that Yu et al.'s CLS scheme [21] was insecure against PKR or MKGC attacks. As a countermeasure, Yuan et al. [23] designed an enhanced scheme, but it did not satisfy strong unforgeability. Based on the Boneh–Boyen signature [38] and Pointcheval–Sanders signature [39], Canard and Trinh [25] constructed a CLS scheme with a low computational cost. However, Canard and Trinh's CLS scheme [25] was existentially unforgeable in the standard model. Subsequently, Huang et al. [22] constructed a CLS scheme with strong unforgeability in the standard model. Unfortunately, Yang et al. [30] demonstrated that Huang et al.'s CLS scheme [22] failed to achieve a strong unforgeability and was vulnerable to MKGC attacks. Furthermore, Yang et al. [30] presented a secure CLS scheme, but their scheme still has some drawbacks, including a longer private key size and a higher computational overhead than those of the previous schemes.

Digital signatures are widely used to ensure data authenticity and integrity. Yeh et al. [4] devised a CLS scheme for IoT environments. However, Jia et al. [40] demonstrated that Yeh et al.'s scheme [4] was insecure against PKR attacks and then proposed a new CLS scheme to overcome the flaws of Yeh et al.'s scheme [4]. Based on technologies such as RSA, DSA, and Merkle tree, Li et al. [41] proposed an IoT data communication framework to provide integrity and authenticity. Frädrich et al. [42] used redactable signature [43] to design another framework for the IoT environment to allow the redaction of parts from signed data and proved its security in the random oracle model. To achieve the security requirements in IoT, Challa et al. [44] presented a new signature-based authenticated key establishment scheme for the IoT environment. Based on Nyberg's fast one-way accumulator [45], Yao et al. [46] designed a lightweight multicast authentication mechanism for small scale IoT applications. Yang et al. [47] proposed a certificateless aggregate signature scheme for vehicular ad hoc networks to reduce transmission bandwidth and verification overhead of signatures. To protect the identity privacy of IoT devices, Yang et al. [48] constructed a strong designated-verifier proxy re-signature (SDVPRS) scheme in the standard model and

applied it to the IoT environment. Unfortunately, the existing data integrity and authenticity schemes in IoT have two drawbacks. (1) Some schemes [41,42,44,46–48] require heavy management and communication overheads of certificates to achieve authenticity authentication of the user's public key. (2) Most of the schemes [4,40–42,44,46,47] are proved to be secure in the random oracle model. To fill thess gaps, Karati et al. [3] presented a secure CLS scheme for IoT environments in the standard model, but their scheme did not consider a strong unforgeability and replay attacks. To our best knowledge, designing an efficient CLS scheme that both satisfies strong unforgeability in the standard model and is resistant to PKR attacks, MKGC attacks, and replay attacks remains an open issue. Therefore, in this paper, we advance such a construction for IoT environments to ensure data integrity and data authenticity.

3. Preliminaries

Here, we briefly review some preliminary knowledge, including the definition of bilinear pairings, the complexity assumptions, and the security model of the CLS schemes.

3.1. Bilinear Paring

Assume that G_1 and G_2 are cyclic groups with the same order of prime p and that g is any generator of G_1. A bilinear pair $e : G_1 \times G_1 \rightarrow G_2$ is a map that satisfies the following conditions [23]:

- Bilinearity: $e(g^a, g^b) = e(g, g)^{ab}$ for all $a, b \in Z_p$.
- Nondegeneracy: $e(g, g) \neq 1$.
- Computability: There is an algorithm that can efficiently calculate $e(g^a, g^b)$ for any $a, b \in Z_p$.

3.2. Complexity Assumptions

Given two elements $g, h \in G_1$, the discrete logarithm (DL) problem [30] is to find an integer $a \in Z_p$ such that $h = g^a$.

Let $\mathcal{A}$ denote an attacker with probabilistic polynomial time (PPT). The advantage ε of $\mathcal{A}$ to solve the DL problem in G_1 is defined as

$$Adv_{\mathcal{A}}^{DL} = \Pr[\mathcal{A}(g, h) = a : a \in Z_p] \geq \varepsilon.$$

Definition 1 (DL assumption)**.** *We say that the DL assumption holds in G_1 if there is no PPT attacker $\mathcal{A}$ to solve the DL problem with a non-negligible advantage ε.*

Given three elements, $g, g^a, and\ g^b \in G_1$ for unknown, randomly chosen $a, b \in Z_p$, the computational Diffie–Hellman (CDH) problem [22] is to calculate $g^{ab} \in G_1$.

The advantage ε that any PPT adversary $\mathcal{A}$ can solve the CDH problem in G_1 is defined as

$$Adv_{\mathcal{A}}^{CDH} = \Pr[\mathcal{A}(g, g^a, g^b) = g^{ab} : a, b \in Z_p] \geq \varepsilon.$$

Definition 2 (CDH assumption)**.** *The CDH assumption holds in G_1 if there is no PPT attacker $\mathcal{A}$ to solve the CDH problem with a non-negligible advantage ε.*

Suppose that $\{H_k\}$ represents a family of hash functions $H_k : \{0,1\}^* \rightarrow \{0,1\}^n$, where n is the length of the output value of H_k and k is an index. Given the index k, the collision resistance of hash function (CRHF) [30] H_k is to find $m_0 \neq m_1$ such that $H_k(m_0) = H_k(m_1)$. The advantage ε of any PPT adversary $\mathcal{A}$ in breaking the collision resistance of H_k is defined as

$$Adv_{\mathcal{A}}^{CRHF} = \Pr[\mathcal{A}(k) = (m_0, m_1) : m_0 \neq m_1, H_k(m_0) = H_k(m_1)] \geq \varepsilon.$$

Definition 3 (CRHF assumption)**.** *A hash family* $\{H_k\}$ *is collision resistant if the advantage* ε *of any PPT adversary* $\mathcal{A}$ *to break the collision resistance of hash function* H_k *is negligible.*

3.3. Security Model of CLS

A CLS scheme consists of six algorithms, as follows:

- **Setup**: This algorithm takes as input a security parameter λ, and it outputs the master secret key msk and system parameters $param$.
- **SetPubKey**: This algorithm takes as input $param$ and an identity ID, and it outputs a secret value usk_{ID} and a public key pk_{ID}.
- **PSKExtract**: This algorithm takes as input $param$, ID, and pk_{ID}, and it returns a partial private key psk_{ID} for identity ID.
- **SetSecKey**: Upon receiving $param$, usk_{ID}, and psk_{ID}, this algorithm outputs a private key sk_{ID}.
- **Sign**: This algorithm takes as input $param$, an identity ID's private key sk_{ID} and public key pk_{ID}, a timestamp T, and a message m, and it returns a signature σ on m.
- **Verify**: Upon receiving $param$, ID, pk_{ID}, T, m, and σ, this algorithm outputs 1 if σ is a valid signature of ID on m with respect to T and pk_{ID}, and it outputs 0 otherwise.

According to the security model for CLS presented in References [15,35], a CLS scheme's security should consider two types of adversaries: type I and type II adversaries. A type I adversary is a PKR attacker who knows the secret value of the targeted entity and who can replace any entity's public key with its own. A type I adversary models an outside attacker who is not capable of possessing the master secret key of the KGC. In contrast, a type II adversary models an honest-but-curious KGC attacker who holds the master secret key of the KGC and generates the partial private key of any entity. However, a type II adversary can neither perform the entity's PKR nor obtain the secret value of the targeted entity. To meet more realistic security requirements, Au et al. [49] presented an enhanced security model in which a type II adversary is viewed as an MKGC attacker. In this case, a malicious KGC can access the master secret key of the KGC and may embed extra trapdoors in the system parameters and the master secret key during the initialization phase of the system. Hence, the type II adversary that we focus on is an MKGC attacker. Here, the security model for a strongly secure CLS scheme is formalized via the following games (denoted Games 1 and 2) between a challenger $\mathcal{C}$ and an adversary $\mathcal{A} \in \{\mathcal{A}_1, \mathcal{A}_2\}$.

Game 1: Executed between a challenger $\mathcal{C}$ and a type I adversary $\mathcal{A}_1$.

- *Initialization*: $\mathcal{C}$ first runs the algorithm **Setup** to obtain the master secret key msk and system parameters $param$. $\mathcal{C}$ then runs the algorithm **SetPubKey** to output the secret value usk^* and corresponding public key pk^* of the targeted entity. Finally, $\mathcal{C}$ sends $param$ and (usk^*, pk^*) to $\mathcal{A}_1$ while keeping msk secret.
- *Queries*: $\mathcal{A}_1$ can adaptively access the following oracles with $\mathcal{C}$.
  - *Public Key Query* $\mathcal{O}_{pk}(ID_i)$: Upon receiving an identity ID_i, $\mathcal{C}$ runs the algorithm **SetPubKey** to obtain a public key pk_i and sends it to $\mathcal{A}_1$.
  - *Public Key Replacement (PKR) Query* $\mathcal{O}_{rep}(ID_i, pk_i')$: Upon receiving such a query, $\mathcal{C}$ finds and replaces the original public key pk_i of identity ID_i with a new public key pk_i'.

- *Partial Private Key Query* $\mathcal{O}_{psk}(ID_i, pk_i)$: Upon receiving an identity ID_i and a public key pk_i, $\mathcal{C}$ runs the algorithm **PSKExtract** to generate a partial private key psk_i and sends it to $\mathcal{A}_1$.
- *Private Key Query* $\mathcal{O}_{sk}(ID_i)$: When $\mathcal{A}_1$ initiates a private key inquiry about an identity ID_i, $\mathcal{C}$ executes the algorithm **SetSecKey** to produce a private key sk_i and sends it to $\mathcal{A}_1$. Note that $\mathcal{C}$ returns the symbol $\perp$ if ID_i has already appeared in PKR queries.
- *Signing Query* $\mathcal{O}_{sign}(ID_i, T, m)$: Upon receiving an identity ID_i, a timestamp Tm and a message m, $\mathcal{C}$ first executes the algorithm **SetSecKey** to produce a private key sk_i and then uses sk_i, T, and the identity ID_i's matching public key pk_i to execute the algorithm **Sign** to produce a signature σ_i on m. Finally, $\mathcal{C}$ sends σ_i to $\mathcal{A}_1$.

- *Forgery*: $\mathcal{A}_1$ eventually outputs a forged signature σ^* on a message m^* corresponding to an identity ID^*, a timestamp T^*, and the targeted public key pk^*. It is said that $\mathcal{A}_1$ wins this game when the following conditions are fulfilled:

  (1) $\mathbf{Verify}(param, ID^*, pk^*, T^*, m^*, \sigma^*) = 1$.
  (2) ID^* is not requested in $\mathcal{O}_{psk}(ID_i, pk_i)$ and $\mathcal{O}_{sk}(ID_i)$.
  (3) $(ID^*, T^*, m^*, \sigma^*)$ is not an output of the oracle $\mathcal{O}_{sign}(ID_i, T, m)$.

Game 2: Executed between a challenger $\mathcal{C}$ and a type II adversary $\mathcal{A}_2$. To launch malicious attacks more easily, $\mathcal{A}_2$ is allowed to set some trapdoors during the initialization phase of the game.

- *Initialization*: $\mathcal{C}$ invokes $\mathcal{A}_2$ to produce the master secret key msk and system parameters $param$. Then, $\mathcal{C}$ runs the algorithm **SetPubKey** to produce the secret value usk^* and the corresponding public key pk^* of the targeted entity. Finally, $\mathcal{C}$ sends pk^* to $\mathcal{A}_2$ while keeping usk^* secret.
- *Queries*: $\mathcal{A}_2$ can adaptively access the oracles $\mathcal{O}_{pk}(ID_i)$, $\mathcal{O}_{sk}(ID_i)$, and $\mathcal{O}_{sign}(ID_i, T, m)$, which are defined in Game 1, and $\mathcal{C}$ responds in the same way as it does in Game 1.
- *Forgery*: $\mathcal{A}_2$ eventually outputs a forged signature σ^* on a message m^* corresponding to an identity ID^*, a timestamp T^*, and the targeted public key pk^*. It is said that $\mathcal{A}_2$ wins this game when the following conditions are fulfilled:

  (1) $\mathbf{Verify}(param, ID^*, pk^*, T^*, m^*, \sigma^*) = 1$.
  (2) ID^* is not requested in $\mathcal{O}_{sk}(ID_i)$.
  (3) $(ID^*, T^*, m^*, \sigma^*)$ is not an output of the oracle $\mathcal{O}_{sign}(ID_i, T, m)$.

$Adv_{\mathcal{A}_i}^{SUF} = \Pr[\mathcal{A}_i \, succeeds]$ denotes the advantage that $\mathcal{A}_i$ wins the above games, where $i \in \{1, 2\}$.

Definition 4. *A CLS scheme is said to be strongly unforgeable against adaptive chosen message attacks if the advantages* $Adv_{\mathcal{A}_1}^{SUF}$ *and* $Adv_{\mathcal{A}_2}^{SUF}$ *are negligible for all PPT type I adversaries* $\mathcal{A}_1$ *and type II adversaries* $\mathcal{A}_2$.

4. Proposed CLS Scheme

Based on Waters' scheme [26] and its variants [28,34], we propose an undeniable and strongly unforgeable CLS scheme in the standard model. Our CLS scheme is described as follows.

- **Setup**: Upon giving the security parameter λ as input, the KGC produces the master secret key and system parameters by performing the following steps.

  (1) Select G_1 and G_2 as two cyclic groups with prime order p, a generator g of G_1, and a bilinear pairing $e : G_1 \times G_1 \to G_2$.
  (2) Select two random values $\alpha, \beta \in Z_p^*$ and compute $g_1 = g^\alpha$ and $g_2 = g^\beta$.

(3) Select two random elements $u_0, v_0 \in G_1$ and two vectors $\vec{u} = (u_i)$ and $\vec{v} = (v_j)$ of lengths n_u and n_m, respectively, where $u_i, v_j \in G_1$ for $i = 1, ..., n_u$ and $j = 1, ..., n_m$.

(4) Select three collision-resistant hash functions $H_1 : \{0,1\}^* \to \{0,1\}^{n_u}$, $H_2 : \{0,1\}^* \to \{0,1\}^{n_m}$, and $H_3 : \{0,1\}^* \to Z_p^*$.

(5) Secretly keep the master key $msk = g^{\alpha\beta}$ and publicly broadcast the system parameters $param = \{G_1, G_2, p, g, e, g_1, g_2, u_0, v_0, \vec{u}, \vec{v}, H_1, H_2, H_3\}$.

- **SetPubKey**: An entity with identity ID randomly selects $\theta_1, \theta_2, \theta_3 \in Z_p^*$ and computes

$$pk_{ID,1} = g^{\theta_1}, \quad pk_{ID,2} = g^{\theta_2} \quad \text{and} \quad pk_{ID,3} = g^{\theta_3}.$$

Then, the entity computes $usk_{ID} = g^{\theta_1\theta_2}$ as its secret value and sets its public key $pk_{ID} = (pk_{ID,1}, pk_{ID,2}, pk_{ID,3})$.

- **PSKExtract**: Given an identity ID and a public key pk_{ID} of an entity, the KGC first computes a vector $\vec{Q} = H_1(ID, pk_{ID}) = (Q_1, ..., Q_{n_u}) \in \{0,1\}^{n_u}$ and $U_{ID} = u_0 \prod_{i=1}^{n_u} u_i^{Q_i}$. Then, the KGC selects $s \in Z_p^*$ at random and computes

$$psk_{ID,1} = g^{\alpha\beta}(U_{ID})^s \quad \text{and} \quad psk_{ID,2} = g^s.$$

Finally, the KGC sends the partial private key $psk_{ID} = (psk_{ID,1}, psk_{ID,2})$ to the entity via a secure channel.

After receiving $psk_{ID} = (psk_{ID,1}, psk_{ID,2})$ from the KGC, the entity can check the correctness of psk_{ID} by verifying

$$e(psk_{ID,1}, g) = e(g_2, g_1)e(U_{ID}, psk_{ID,2}).$$

If this equation holds, then the entity accepts psk_{ID} as a valid partial private key.

- **SetSecKey**: The entity with identity ID selects a random value $r \in Z_p^*$ and computes a vector $\vec{Q} = H_1(ID, pk_{ID}) = (Q_1, ..., Q_{n_u}) \in \{0,1\}^{n_u}$ and $U_{ID} = u_0 \prod_{i=1}^{n_u} u_i^{Q_i}$, where pk_{ID} is ID's public key. Then, the entity uses its secret value usk_{ID} and partial private key $psk_{ID} = (psk_{ID,1}, psk_{ID,2})$ to compute its private key

$$\begin{aligned} sk_{ID} &= (sk_{ID,1}, sk_{ID,2}) \\ &= (psk_{ID,1} \cdot usk_{ID} \cdot (U_{ID})^r, psk_{ID,2} \cdot g^r) \\ &= (g^{\alpha\beta}(U_{ID})^s \cdot g^{\theta_1\theta_2} \cdot (U_{ID})^r, g^s \cdot g^r) \\ &= (g^{\alpha\beta} g^{\theta_1\theta_2} (U_{ID})^{s+r}, g^{s+r}). \end{aligned}$$

- **Sign**: The signer with identity ID generates a signature of a message m by performing the following steps.

(1) Select a random value $r_m \in Z_p^*$ and compute $\sigma_3 = g^{r_m}$.

(2) Choose the current timestamp T and compute a vector $\vec{M} = H_2(m, T) = (M_1, ..., M_{n_m}) \in \{0,1\}^{n_m}$ and $V_m = v_0 \prod_{j=1}^{n_m} v_j^{M_j}$.

(3) Compute

$$h = H_3(m, T, ID, pk_{ID}, sk_{ID,2}, \sigma_3, param),$$
$$\sigma_1 = sk_{ID,1} \cdot ((pk_{ID,3})^h V_m)^{r_m},$$
$$\sigma_2 = sk_{ID,2},$$

where $sk_{ID} = (sk_{ID,1}, sk_{ID,2})$ and $pk_{ID} = (pk_{ID,1}, pk_{ID,2}, pk_{ID,3})$ are the private and public keys of identity ID, respectively.

(4) Output $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ as a signature of m.

- **Verify**: Given the signer's identity ID and public key $pk_{ID} = (pk_{ID,1}, pk_{ID,2}, pk_{ID,3})$, timestamp T, and a signature $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ of message m, the verifier first chooses the current time $T^{'}$. Then, the verifier verifies the legality of σ as follows.

(1) If $T^{'} - T > \delta$, where δ is a threshold value, the verifier refuses to verify the validity of σ and exits.

(2) If $T^{'} - T \leq \delta$, the verifier computes $\vec{Q} = H_1(ID, pk_{ID})$, U_{ID}, $\vec{M} = H_2(m, T)$, V_m and

$$h = H_3(m, T, ID, pk_{ID}, sk_{ID,2}, \sigma_3, param).$$

Then, the verifier checks

$$e(\sigma_1, g) = e(g_2, g_1) \cdot e(pk_{ID,1}, pk_{ID,2}) \cdot e(U_{ID}, \sigma_2) \cdot e((pk_{ID,3})^h V_m, \sigma_3).$$

If this equation holds, the verifier accepts σ and outputs 1; otherwise, the verifier rejects σ and outputs 0.

Correctness: The correctness of a signature $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ on a message m is presented as follows:

$$\begin{aligned} e(\sigma_1, g) =& e(sk_{ID,1} \cdot ((pk_{ID,3})^h V_m)^{r_m}, g) \\ =& e(g^{\alpha\beta} g^{\theta_1\theta_2} (U_{ID})^{s+r} \cdot ((pk_{ID,3})^h V_m)^{r_m}, g) \\ =& e(g^{\alpha\beta}, g) \cdot e(g^{\theta_1\theta_2}, g) \cdot e((U_{ID})^{s+r}, g) \cdot e(((pk_{ID,3})^h V_m)^{r_m}, g) \\ =& e(g^{\alpha}, g^{\beta}) \cdot e(g^{\theta_1}, g^{\theta_2}) \cdot e(U_{ID}, g^{s+r}) \cdot e((pk_{ID,3})^h V_m, g^{r_m}) \\ =& e(g_2, g_1) \cdot e(pk_{ID,1}, pk_{ID,2}) \cdot e(U_{ID}, \sigma_2) \cdot e((pk_{ID,3})^h V_m, \sigma_3). \end{aligned}$$

5. Security Proof

In our CLS scheme, the algorithm **SetSecKey** randomizes the entity's secret value usk_{ID} and partial private key $psk_{ID} = (psk_{ID,1}, psk_{ID,2})$ to generate the final private key $sk_{ID} = (sk_{ID,1}, sk_{ID,2}) = (psk_{ID,1} \cdot usk_{ID} \cdot (U_{ID})^r, psk_{ID,2} \cdot g^r)$. Hence, it is not feasible for a malicious KGC to produce a valid signature without the secret value usk_{ID}. Additionally, the KGC cannot derive the entity's private key sk_{ID} from the master secret key msk and the entity's partial private key psk_{ID}.

To prevent PKR and MKGC attacks, a part $pk_{ID,3}$ of the entity's public key is embedded in the signature σ. Only each entity can produce its legal public key $pk_{ID} = (pk_{ID,1}, pk_{ID,2}, pk_{ID,3}) = (g^{\theta_1}, g^{\theta_2}, g^{\theta_3})$; thus, the malicious KGC can neither set the entity's public key at will nor derive the secret value usk_{ID} from the signature. Furthermore, it is impossible to obtain the value $(pk_{ID,3})^{r_m}$ directly from the entity's public key pk_{ID} and the public value $\sigma_3 = g^{r_m}$ of the signature unless the adversary can solve the CDH problem.

The algorithm **PSKExtract** binds each entity's public key pk_{ID}, identity ID, and partial private key psk_{ID}, which can enhance the trust level of the proposed CLS scheme. If the KGC attempts to replace the entity's public key pk_{ID}, then the entity's identity ID and the new public key must be re-bound to compute a new partial private key, which results in the entity's identity ID corresponding to two public keys and two partial private keys. Therefore, our CLS scheme can easily determine whether the KGC replaced the entity's public key.

In addition, H_3 is a collision-resistant hash function. The hash value h combines the message m, the identity ID, public key pk_{ID}, two values σ_2, and σ_3 in the signature, a timestamp T, and system parameters $param$ as $h = H_3(m, T, ID, pk_{ID}, sk_{ID,2}, \sigma_3, param)$. Hence, an attacker cannot forge a new valid signature from an existing signature on a message; that is, an adversary cannot generate a valid signature on any previously signed/new message in our CLS scheme.

In the following, we introduce two theorems to demonstrate that our CLS scheme satisfies a strong unforgeability against PKR and MKGC attacks in the standard model. Reduction technology is used to prove the strong unforgeability of the proposed scheme; specifically, if an attacker breaks the security of the scheme, a solver then uses the attacker's ability to solve the underlying hard problem related to the scheme. However, this problem is intractable in reality; thus, such an attacker does not exist. Furthermore, we prove that the proposed CLS scheme can resist replay attacks.

Theorem 1. *In the standard model, our CLS scheme is strongly unforgeable against PKR attacks. Specifically, there is a type I adversary $\mathcal{A}_1$ that breaks the security of the proposed CLS scheme with advantage ε_1 after making at most q_{pk} public key queries, q_{psk} partial private key queries, q_{rep} PKR queries, q_{sk} private key queries, and q_s signing queries. Then, an algorithm $\mathcal{C}$ can use the forgery of $\mathcal{A}_1$ to solve the CDH problem with advantage $\varepsilon_1^{'}$.*

Proof. $\mathcal{C}$ is given a random instance $(g, g^a, g^b) \in G_1^3$ of the CDH problem, and $\mathcal{C}$'s goal is to output g^{ab} with the help of $\mathcal{A}_1$. The algorithm $\mathcal{C}$ simulates the challenger in Game 1 and responds to $\mathcal{A}_1$'s queries as follows.

- *Initialization*: $\mathcal{C}$ first sets $l_u = 2(q_{psk} + q_{sk} + q_s)$ and $l_m = 2q_s$ such that $l_u(n_u + 1) < p$ and $l_m(n_m + 1) < p$. Then, $\mathcal{C}$ simulates the algorithm **Setup** by performing the following steps:

  (1) Randomly select $k_u (0 \le k_u \le n_u)$ and $k_m (0 \le k_m \le n_m)$.
  (2) Randomly select $x_0, x_1, ..., x_{n_u} \in Z_{l_u}$, $y_0, y_1, ..., y_{n_u} \in Z_p$, $c_0, c_1, ..., c_{n_m} \in Z_{l_m}$, and $d_0, d_1, ..., d_{n_m} \in Z_p$.
  (3) Select three hash functions $H_1 : \{0,1\}^* \to \{0,1\}^{n_u}$, $H_2 : \{0,1\}^* \to \{0,1\}^{n_m}$, and $H_3 : \{0,1\}^* \to Z_p^*$. Note that the adopted hash functions are not considered to be random oracles in the following proof.
  (4) Set $g_1 = g^a$ and $g_2 = g^b$, where g^a and g^b are from the input of the instance of the CDH problem. Note that the master secret key is implicitly set to $msk = g^{ab}$.
  (5) Assign $u_0 = g_2^{-l_u k_u + x_0} g^{y_0}$, $u_i = g_2^{x_i} g^{y_i}$ for $i = 1, ..., n_u$, $v_0 = g_2^{-l_m k_m + c_0} g^{d_0}$, and $v_j = g_2^{c_j} g^{d_j}$ for $j = 1, ..., n_m$, and set $\vec{u} = (u_1, ..., u_{n_u})$ and $\vec{v} = (v_1, ..., v_{n_m})$.
  (6) Select three random integers $\theta_1^*, \theta_2^*, \theta_3^* \in Z_p^*$ and compute $pk_1^* = g^{\theta_1^*}$, $pk_2^* = g^{\theta_2^*}$, and $pk_3^* = g^{\theta_3^*}$. Next, set the secret value of the targeted entity to $usk^* = g^{\theta_1^* \theta_2^*}$ and the corresponding public key to $pk^* = (pk_1^*, pk_2^*, pk_3^*)$.
  (7) Send system parameters $param = \{G_1, G_2, p, g, e, g_1, g_2, u_0, v_0, \vec{u}, \vec{v}, H_1, H_2, H_3\}$ and the targeted entity's secret value/public key pair (usk^*, pk^*) to $\mathcal{A}_1$.

From the perspective of $\mathcal{A}_1$, the distribution of the system parameters produced by $\mathcal{C}$ is identical to the real construction.

In our CLS scheme, we have $\vec{Q} = H_1(ID, pk_{ID}) = (Q_1, ..., Q_{n_u}) \in \{0,1\}^{n_u}$ for an identity ID and a public key pk_{ID}, and we have $\vec{M} = H_2(m, T) = (M_1, ..., M_{n_m}) \in \{0,1\}^{n_m}$ for a message m and a timestamp T. Aiming to simplify the analysis, we define the following four functions:

$$F(ID) = -l_u k_u + x_0 + \sum_{i=1}^{n_u} x_i Q_i, J(ID) = y_0 + \sum_{i=1}^{n_u} y_i Q_i,$$
$$K(m) = -l_m k_m + c_0 + \sum_{j=1}^{n_m} c_j M_j, L(m) = d_0 + \sum_{j=1}^{n_m} d_j M_j.$$

Hence, we have the following equations:

$$U_{ID} = u_0 \prod_{i=1}^{n_u} u_i^{Q_i} = g_2^{F(ID)} g^{J(ID)},$$
$$V_m = v_0 \prod_{j=1}^{n_m} v_j^{M_j} = g_2^{K(m)} g^{L(m)}.$$

- *Queries*: $\mathcal{C}$ maintains a list $\mathcal{L} = \{(ID_i, \theta_{i,1}, \theta_{i,2}, \theta_{i,3}, usk_i, pk_i, psk_i, sk_i)\}$, which is initially empty. $\mathcal{C}$ constructs the following oracles to answer a series of $\mathcal{A}_1$'s queries.
  - *Public Key Query* $\mathcal{O}_{pk}(ID_i)$: When $\mathcal{A}_1$ initiates such an inquiry for an identity ID_i, $\mathcal{C}$ looks up the corresponding entry in the list $\mathcal{L}$. If ID_i is found in $\mathcal{L}$, $\mathcal{C}$ returns pk_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly selects $\theta_{i,1}, \theta_{i,2}, \theta_{i,3} \in Z_p^*$ and computes the secret value $usk_i = g^{\theta_{i,1}\theta_{i,2}}$ and the public key $pk_i = (pk_{i,1}, pk_{i,2}, pk_{i,3}) = (g^{\theta_{i,1}}, g^{\theta_{i,2}}, g^{\theta_{i,3}})$. Then, $\mathcal{C}$ stores $\{(ID_i, \theta_{i,1}, \theta_{i,2}, \theta_{i,3}, usk_i, pk_i)\}$ in $\mathcal{L}$ and sends pk_i to $\mathcal{A}_1$.
  - *Public Key Replacement Query* $\mathcal{O}_{rep}(ID_i, pk_i')$: If there is an entry for the identity ID_i in the list $\mathcal{L}$, $\mathcal{C}$ replaces the original public key pk_i of ID_i with a new public key pk_i'. Otherwise, $\mathcal{C}$ directly sets pk_i' as the public key of ID_i.
  - *Partial Private Key Query* $\mathcal{O}_{psk}(ID_i, pk_i)$: When $\mathcal{A}_1$ requests a partial private key of an identity ID_i and a public key pk_i, $\mathcal{C}$ returns psk_i to $\mathcal{A}_1$ if there is an entry for ID_i and pk_i in the list $\mathcal{L}$. Otherwise, $\mathcal{C}$ computes $F(ID_i)$ and $J(ID_i)$.

    (1) If $F(ID_i) \neq 0 \bmod l_u$, $\mathcal{C}$ randomly selects $s_i \in Z_p^*$ and calculates a partial private key

    $$psk_i = (psk_{i,1}, psk_{i,2})$$
    $$= (g_1^{\frac{-J(ID_i)}{F(ID_i)}} (U_{ID_i})^{s_i}, g_1^{\frac{-1}{F(ID_i)}} g^{s_i}),$$

    where $U_{ID_i} = u_0 \prod_{k=1}^{n_u} u_k^{Q_{i,k}}$ and $\vec{Q}_i = H_1(ID_i, pk_i) = (Q_{i,1}, ..., Q_{i,n_u}) \in \{0,1\}^{n_u}$. Then, $\mathcal{C}$ stores the partial private key of the corresponding entry in $\mathcal{L}$ and sends psk_i to $\mathcal{A}_1$.

    (2) If $F(ID_i) = 0 \bmod l_u$, $\mathcal{C}$ terminates the simulation.

Note that the partial private key $psk_i = (psk_{i,1}, psk_{i,2})$ generated by $\mathcal{C}$ is legal.

$$\begin{aligned} psk_{i,1} &= g_1^{\frac{-J(ID_i)}{F(ID_i)}} (U_{ID_i})^{s_i} \\ &= g_2^a (g_2^{F(ID_i)} g^{J(ID_i)})^{\frac{-a}{F(ID_i)}} (U_{ID_i})^{s_i} \\ &= g_2^a (U_{ID_i})^{s_i - \frac{a}{F(ID_i)}} \\ &= g^{ab} (U_{ID_i})^{s_i - \frac{a}{F(ID_i)}}, \\ psk_{i,2} &= g_1^{\frac{-1}{F(ID_i)}} g^{s_i} = g^{s_i - \frac{a}{F(ID_i)}}. \end{aligned}$$

Then, we have

$$e(psk_{i,1}, g) = e(g_2, g_1) e(U_{ID_i}, psk_{i,2}).$$

Hence, from $\mathcal{A}_1$'s perspective, the partial private key psk_i simulated by $\mathcal{C}$ is computationally indistinguishable from that computed by the real KGC.

- *Private Key Query* $\mathcal{O}_{sk}(ID_i)$: When $\mathcal{A}_1$ requests the private key of an identity ID_i, $\mathcal{C}$ checks for an entry of ID_i in $\mathcal{L}$. If it exists, $\mathcal{C}$ returns sk_i to $\mathcal{A}_1$; otherwise, $\mathcal{C}$ computes $F(ID_i)$ and $J(ID_i)$. If $F(ID_i) = 0 \bmod l_u$, $\mathcal{C}$ terminates; otherwise, $\mathcal{C}$ initiates a public key query about ID_i to acquire a secret value usk_i and a public key pk_i and then initiates a partial private key query with (ID_i, pk_i) to acquire a partial private key psk_i. Next, $\mathcal{C}$ executes the algorithm **SetSecKey** to create a private key sk_i, stores sk_i of the corresponding entry in $\mathcal{L}$ and sends sk_i to $\mathcal{A}_1$.
- *Signing Query* $\mathcal{O}_{sign}(ID_i, T, m)$: Upon receiving an identity ID_i, a timestamp T, and a message m, $\mathcal{C}$ issues a query $\mathcal{O}_{pk}(ID_i)$ to acquire a public key $pk_i = (pk_{i,1}, pk_{i,2}, pk_{i,3})$ and the triplet $(\theta_{i,1}, \theta_{i,2}, \theta_{i,3})$. Then, $\mathcal{C}$ proceeds as follows.

  (1) If $F(ID_i) \neq 0 \bmod l_u$, $\mathcal{C}$ first makes a query $\mathcal{O}_{sk}(ID_i)$ to acquire a private key sk_i and then runs the algorithm **Sign** to generate a signature σ_i of m. Finally, $\mathcal{C}$ sends σ_i to $\mathcal{A}_1$.

  (2) If $F(ID_i) = 0 \bmod l_u$, $\mathcal{C}$ computes $K(m)$. If $K(m) = 0 \bmod l_m$, $\mathcal{C}$ terminates; otherwise, $\mathcal{C}$ randomly selects $r_i, r_m \in Z_p^*$ and computes $\vec{Q}_i = H_1(ID_i, pk_i)$, U_{ID_i}, $\vec{M} = H_2(m, T)$, and V_m. Furthermore, $\mathcal{C}$ computes

$$\begin{aligned} \sigma_{i,2} &= g^{r_i}, \\ \sigma_{i,3} &= g_1^{\frac{-1}{K(m)}} g^{r_m}, \\ h_i &= H_3(m, T, ID_i, pk_i, \sigma_{i,2}, \sigma_{i,3}, param), \\ \sigma_{i,1} &= (U_{ID_i})^{r_i} g_1^{\frac{-L(m) - h_i \theta_{i,3}}{K(m)}} ((pk_{i,3})^{h_i} V_m)^{r_m} g^{\theta_{i,1} \theta_{i,2}}. \end{aligned}$$

  Finally, $\mathcal{C}$ sends $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, \sigma_{i,3})$ to $\mathcal{A}_1$.

For $\tilde{r}_m = r_m - \frac{a}{K(m)}$, we have

$$\begin{aligned}
\sigma_{i,1} =&(U_{ID_i})^{r_i} g_1^{\frac{-L(m)-h_i\theta_{i,3}}{K(m)}} ((pk_{i,3})^{h_i} V_m)^{r_m} g^{\theta_{i,1}\theta_{i,2}} \\
=&(U_{ID_i})^{r_i} g^{ab} (g_2^{K(m)} g^{L(m)} g^{h_i\theta_{i,3}})^{\frac{-a}{K(m)}} ((pk_{i,3})^{h_i} V_m)^{r_m} g^{\theta_{i,1}\theta_{i,2}} \\
=&g^{ab} g^{\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{r_m - \frac{a}{K(m)}} \\
=&g^{ab} g^{\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{\tilde{r}_m}, \\
\sigma_{i,2} =&g^{r_i}, \\
\sigma_{i,3} =&g_1^{\frac{-1}{K(m)}} g^{r_m} = g^{r_m - \frac{a}{K(m)}} = g^{\tilde{r}_m}.
\end{aligned}$$

Clearly, the signature $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, \sigma_{i,3})$ generated by $\mathcal{C}$ is legal because σ_i satisfies the following verification equation:

$$\begin{aligned}
e(\sigma_{i,1}, g) =&e(g^{ab} g^{\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{\tilde{r}_m}, g) \\
=&e(g^a, g^b) e(g^{\theta_{i,1}}, g^{\theta_{i,2}}) e(U_{ID_i}, g^{r_i}) e((pk_{i,3})^{h_i} V_m, g^{\tilde{r}_m}) \\
=&e(g_2, g_1) \cdot e(pk_{i,1}, pk_{i,2}) \cdot e(U_{ID_i}, \sigma_{i,2}) e((pk_{i,3})^{h_i} V_m, \sigma_{i,3}).
\end{aligned}$$

From $\mathcal{A}_1$'s perspective, the signatures simulated by $\mathcal{C}$ are computationally indistinguishable from those produced by the real signer.

- *Forgery*: $\mathcal{A}_1$ eventually outputs a signature $\sigma^* = (\sigma_1^*, \sigma_2^*, \sigma_3^*)$ on a message m^* corresponding to an identity ID^*, a timestamp T^*, and targeted public key pk^*. If $F(ID^*) \neq 0 \bmod p$ or $K(m^*) \neq 0 \bmod p$, $\mathcal{C}$ terminates; otherwise, $\mathcal{C}$ computes $h^* = H_3(m^*, T^*, ID^*, pk^*, \sigma_2^*, \sigma_3^*, param)$ and uses $(\theta_1^*, \theta_2^*, \theta_3^*)$ to output g^{ab} as a solution to the CDH instance as follows:

$$\begin{aligned}
&\frac{\sigma_1^*}{g^{\theta_1^*\theta_2^*} (\sigma_2^*)^{J(ID^*)} (\sigma_3^*)^{L(m^*)+h^*\theta_3^*}} \\
=&\frac{g_2^a g^{\theta_1^*\theta_2^*} (U_{ID^*})^{r_{ID}^*} ((pk_3^*)^{h^*} V_{m^*})^{r_m^*}}{g^{\theta_1^*\theta_2^*} (g^{r_{ID}^*})^{J(ID^*)} (g^{r_m^*})^{L(m^*)+h^*\theta_3^*}} \\
=&\frac{g_2^a (g_2^{F(ID^*)} g^{J(ID^*)})^{r_{ID}^*} ((g^{\theta_3^*})^{h^*} (g_2^{K(m^*)} g^{L(m^*)}))^{r_m^*}}{(g^{J(ID^*)})^{r_{ID}^*} (g^{\theta_3^*})^{r_m^* h^*} (g^{L(m^*)})^{r_m^*}} \\
=&g_2^a g_2^{F(ID^*) r_{ID}^*} g_2^{K(m^*) r_m^*} \\
&\text{(since} \quad F(ID^*) = K(m^*) = 0 \bmod p) \\
=&g_2^a \\
=&g^{ab}.
\end{aligned}$$

Now, we analyze the probability that $\mathcal{C}$ can successfully solve the CDH problem. If the following conditions hold, $\mathcal{C}$ completes the above simulation without aborting.

(1) All partial private key queries on (ID_i, pk_i) have $F(ID_i) \neq 0 \bmod l_u$.
(2) All private key queries on ID_i have $F(ID_i) \neq 0 \bmod l_u$.
(3) All signing queries on (ID_i, T, m) have $F(ID_i) \neq 0 \bmod l_u$ or $K(m) \neq 0 \bmod l_m$.

(4) In the forgery phase, $F(ID^*) = 0 \bmod p$ and $K(m^*) = 0 \bmod p$.

Here, we define four independent events X_i, X^*, Y_j, and Y^* as follows.

- $X_i : F(ID_i) \neq 0 \bmod l_u$ for the ith query, where $1 \leq i \leq q_{psk} + q_{sk} + q_s$.
- $X^* : F(ID^*) = 0 \bmod p$.
- $Y_j : K(m_j) \neq 0 \bmod l_m$ for the jth query, where $1 \leq j \leq q_s$.
- $Y^* : K(m^*) = 0 \bmod p$.

Because the events $\bigcap_{i=1}^{q_{psk}+q_{sk}+q_s} X_i$, X^*, $\bigcap_{j=1}^{q_s} Y_j$, and Y^* are independent, the probability that $\mathcal{C}$ does not terminate is

$$\begin{aligned}\Pr[\neg abort] &\geq \Pr[\bigcap_{i=1}^{q_{psk}+q_{sk}+q_s} X_i \cap X^* \cap \bigcap_{j=1}^{q_s} Y_j \cap Y^*] \\ &= \Pr[X^*] \cdot \Pr[\bigcap_{i=1}^{q_{psk}+q_{sk}+q_s} X_i | X^*] \cdot \Pr[Y^*] \cdot \Pr[\bigcap_{j=1}^{q_s} Y_j | Y^*].\end{aligned}$$

From $l_u(n_u + 1) < p$, $l_m(n_m + 1) < p$, $0 \leq k_u \leq n_u$, $0 \leq k_m \leq n_m$, $x_0, x_1, ..., x_{n_u} \in Z_{l_u}$ and $c_0, c_1, ..., c_{n_m} \in Z_{l_m}$, we have $0 \leq l_u k_u < p$, $0 \leq l_m k_m < p$, $0 \leq x_0 + \sum_{i=1}^{n_u} x_i Q_i < p$ and $0 \leq c_0 + \sum_{j=1}^{n_m} c_j M_j < p$. Therefore, it is easy to derive $F(ID) = 0 \bmod l_u$ and $K(m) = 0 \bmod l_m$ from $F(ID) = 0 \bmod p$ and $K(m) = 0 \bmod p$, respectively. Moreover, $F(ID) \neq 0 \bmod l_u$ implies that $F(ID) \neq 0 \bmod p$, and $K(m) \neq 0 \bmod l_m$ implies that $K(m) \neq 0 \bmod p$. Since $x_0, x_1, ..., x_{n_u}$ and $c_0, c_1, ..., c_{n_m}$ are randomly chosen, we obtain the probabilities of the events X^* and Y^* as follows:

$$\begin{aligned}\Pr[X^*] &= \Pr[F(ID^*) = 0 \bmod p] \\ &\geq \Pr[F(ID^*) = 0 \bmod p \cap F(ID^*) = 0 \bmod l_u] \\ &= \Pr[F(ID^*) = 0 \bmod l_u] \Pr[F(ID^*) = 0 \bmod p | F(ID^*) = 0 \bmod l_u] \\ &= \frac{1}{l_u} \frac{1}{n_u + 1}, \\ \Pr[Y^*] &= \Pr[K(m^*) = 0 \bmod p] \\ &\geq \Pr[K(m^*) = 0 \bmod p \cap K(m^*) = 0 \bmod l_m] \\ &= \Pr[K(m^*) = 0 \bmod l_m] \Pr[K(m^*) = 0 \bmod p | K(m^*) = 0 \bmod l_m] \\ &= \frac{1}{l_m} \frac{1}{n_m + 1}.\end{aligned}$$

Furthermore, we have

$$\begin{aligned}\Pr[\bigcap_{i=1}^{q_{psk}+q_{sk}+q_s} X_i|X^*] =& 1-\Pr[\bigcup_{i=1}^{q_{psk}+q_{sk}+q_s} \neg X_i|X^*]\\ \geq& 1-\sum_{i=1}^{q_{psk}+q_{sk}+q_s}\Pr[\neg X_i|X^*]\\ =& 1-\frac{q_{psk}+q_{sk}+q_s}{l_u},\\ \Pr[\bigcap_{j=1}^{q_s} Y_j|Y^*] =& 1-\Pr[\bigcup_{j=1}^{q_s}\neg Y_j|Y^*]\\ \geq& 1-\sum_{j=1}^{q_s}\Pr[\neg Y_j|Y^*] = 1-\frac{q_s}{l_m}.\end{aligned}$$

Since $l_u = 2(q_{psk}+q_{sk}+q_s)$ and $l_m = 2q_s$, we write

$$\begin{aligned}\Pr[\neg abort] \geq& \frac{1}{l_u}\cdot\frac{1}{n_u+1}\cdot\left(1-\frac{q_{psk}+q_{sk}+q_s}{l_u}\right)\frac{1}{l_m}\cdot\frac{1}{n_m+1}\cdot\left(1-\frac{q_s}{l_m}\right)\\ =& \frac{1}{16q_s(n_u+1)(n_m+1)(q_{psk}+q_{sk}+q_s)}.\end{aligned}$$

Therefore, if $\mathcal{A}_1$ breaks the strong unforgeability of the proposed CLS scheme with advantage ε_1, then $\mathcal{C}$ has an advantage $\varepsilon_1^{'} \geq \frac{\varepsilon_1}{16q_s(n_u+1)(n_m+1)(q_{psk}+q_{sk}+q_s)}$ to solve the given instance of the CDH problem. □

Theorem 2. *In the standard model, the proposed CLS scheme is strongly unforgeable against MKGC attacks launched by the type II adversary $\mathcal{A}_2$. Concretely, assuming that $\mathcal{A}_2$ compromises the security of our CLS scheme with advantage ε_2 after making at most q_{pk} public key queries, q_{sk} private key queries, and q_s signing queries, then an algorithm $\mathcal{C}$ can use $\mathcal{A}_2$'s forgery to solve the CDH problem in G_1 with advantage $\varepsilon_2^{'}$.*

Proof. Supposing that a PPT adversary $\mathcal{A}_2$ breaks the strong unforgeability of our CLS scheme in an adaptive chosen-message attack, we can construct an algorithm $\mathcal{C}$ that calls $\mathcal{A}_2$ as a subroutine to violate the CDH assumption. Assuming that $\mathcal{C}$ is given a random instance $(g, A = g^a, B = g^b) \in G_1^3$, to calculate g^{ab}, $\mathcal{C}$ simulates the challenger in Game 2 to answer all $\mathcal{A}_2$'s queries.

- *Initialization*: For the given values q_{sk}, q_s, n_u, and n_m, $\mathcal{C}$ sets $l_u = 2(q_{sk}+q_s)$ and $l_m = 2q_s$ such that $l_u(n_u+1) < p$ and $l_m(n_m+1) < p$. $\mathcal{C}$ selects a random element $\theta^* \in Z_p^*$ and calculates $pk_3^* = g^{\theta^*}$. Then, $\mathcal{C}$ sets the targeted entity's public key $pk^* = (pk_1^*, pk_2^*, pk_3^*) = (A = g^a, B = g^b, g^{\theta^*})$ and sends parameters (G_1, G_2, p, g, e) and pk^* to $\mathcal{A}_2$.

  Subsequently, $\mathcal{A}_2$ performs the following steps to produce other system parameters and the master secret key.

  (1) Select two random integers k_u and k_m, where $0 \leq k_u \leq n_u$ and $0 \leq k_m \leq n_m$.
  (2) Randomly select $x_0, x_1, ..., x_{n_u} \in Z_{l_u}$, $c_0, c_1, ..., c_{n_m} \in Z_{l_m}$ and $y_0, y_1, ..., y_{n_u}, d_0, d_1, ..., d_{n_m} \in Z_p$.
  (3) Select three collision-resistant hash functions $H_1 : \{0,1\}^* \to \{0,1\}^{n_u}$, $H_2 : \{0,1\}^* \to \{0,1\}^{n_m}$, and $H_3 : \{0,1\}^* \to Z_p^*$.

(4) Assign $u_0 = B^{-l_u k_u + x_0} g^{y_0}$, $u_i = B^{x_i} g^{y_i}$ for $i = 1, ..., n_u$, $v_0 = B^{-l_m k_m + c_0} g^{d_0}$, and $v_j = B^{c_j} g^{d_j}$ for $j = 1, ..., n_m$ and set $\vec{u} = (u_1, ..., u_{n_u})$ and $\vec{v} = (v_1, ..., v_{n_m})$.

(5) Select two random values $\alpha, \beta \in Z_p^*$ and compute $g_1 = g^\alpha$, $g_2 = g^\beta$ and $msk = g^{\alpha\beta}$.

(6) Send parameters $(g_1, g_2, u_0, v_0, \vec{u}, \vec{v}, H_1, H_2, H_3)$ and the master secret key msk to $\mathcal{C}$.

Note that the secret value of the targeted entity is $usk^* = g^{ab}$, which is unknown to $\mathcal{C}$, and the system parameters are $param = \{G_1, G_2, p, g, e, g_1, g_2, u_0, v_0, \vec{u}, \vec{v}, H_1, H_2, H_3\}$.

As the initialization phase in Theorem 1, we define the following four functions:

$$F(ID) = -l_u k_u + x_0 + \sum_{i=1}^{n_u} x_i Q_i, J(ID) = y_0 + \sum_{i=1}^{n_u} y_i Q_i,$$
$$K(m) = -l_m k_m + c_0 + \sum_{j=1}^{n_m} c_j M_j, L(m) = d_0 + \sum_{j=1}^{n_m} d_j M_j.$$

Furthermore, we have the following equations:

$$U_{ID} = u_0 \prod_{i=1}^{n_u} u_i^{Q_i} = B^{F(ID)} g^{J(ID)},$$
$$V_m = v_0 \prod_{j=1}^{n_m} v_j^{M_j} = B^{K(m)} g^{L(m)}.$$

- *Queries*: $\mathcal{C}$ maintains an initially empty list $\mathcal{L}_2$ of tuples $\{(ID_i, \theta_{i,1}, \theta_{i,2}, \theta_{i,3}, pk_i, sk_i)\}$ and builds the following oracles to answer the queries initiated by $\mathcal{A}_2$.
  - *Public Key Query* $\mathcal{O}_{pk}(ID_i)$: When $\mathcal{A}_2$ issues such a query on an identity ID_i, $\mathcal{C}$ looks up the corresponding entry in list $\mathcal{L}_2$ and sends pk_i to $\mathcal{A}_2$. Otherwise, if $\mathcal{L}_2$ does not store this entry, $\mathcal{C}$ randomly selects $\theta_{i,1}, \theta_{i,2}, \theta_{i,3} \in Z_p^*$ and computes the public key $pk_i = (pk_{i,1}, pk_{i,2}, pk_{i,3}) = (A^{\theta_{i,1}}, B^{\theta_{i,2}}, g^{\theta_{i,3}}) = (g^{a\theta_{i,1}}, g^{b\theta_{i,2}}, g^{\theta_{i,3}})$. Note that the secret value is $usk_i = g^{ab\theta_{i,1}\theta_{i,2}}$, but a and b are unknown to $\mathcal{C}$. Then, $\mathcal{C}$ stores $\{(ID_i, \theta_{i,1}, \theta_{i,2}, \theta_{i,3}, pk_i)\}$ in $\mathcal{L}_2$ and transmits pk_i to $\mathcal{A}_2$.
  - *Private Key Query* $\mathcal{O}_{sk}(ID_i)$: Upon receipt of a query on an identity ID_i, $\mathcal{C}$ returns sk_i to $\mathcal{A}_2$ if ID_i is found in $\mathcal{L}_2$; otherwise, $\mathcal{C}$ makes a query $\mathcal{O}_{pk}(ID_i)$ to obtain a public key $pk_i = (pk_{i,1}, pk_{i,2}, pk_{i,3})$ and the triplet $(\theta_{i,1}, \theta_{i,2}, \theta_{i,3})$ and then verifies whether $F(ID_i) = 0 \bmod l_u$.

    (1) If $F(ID_i) = 0 \bmod l_u$, $\mathcal{C}$ exits the simulation.

    (2) If $F(ID_i) \neq 0 \bmod l_u$, $\mathcal{C}$ selects $s_i \in Z_p^*$ and uses the master secret key $msk = g^{\alpha\beta}$ to compute

$$sk_{i,1} = A^{\frac{-J(ID_i)\theta_{i,1}\theta_{i,2}}{F(ID_i)}} g^{\alpha\beta} (U_{ID_i})^{s_i},$$
$$sk_{i,2} = A^{\frac{-\theta_{i,1}\theta_{i,2}}{F(ID_i)}} g^{s_i},$$

    where $U_{ID_i} = u_0 \prod_{k=1}^{n_u} u_k^{Q_{i,k}}$ and $\vec{Q}_i = H_1(ID_i, pk_i) = (Q_{i,1}, ..., Q_{i,n_u}) \in \{0,1\}^{n_u}$. Then, $\mathcal{C}$ stores the private key of the corresponding entry in $\mathcal{L}_2$ and sends $sk_i = (sk_{i,1}, sk_{i,2})$ to $\mathcal{A}_2$.

    The correctness of sk_i simulated by $\mathcal{C}$ is

$$\begin{aligned}
sk_{i,1} =& A^{\frac{-J(ID_i)\theta_{i,1}\theta_{i,2}}{F(ID_i)}} g^{\alpha\beta}(U_{ID_i})^{s_i} \\
=& g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} (g^{bF(ID_i)} g^{J(ID_i)})^{\frac{-a\theta_{i,1}\theta_{i,2}}{F(ID_i)}} (U_{ID_i})^{s_i} \\
=& g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{s_i - \frac{a\theta_{i,1}\theta_{i,2}}{F(ID_i)}} \\
=& g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{\tilde{s}_i}, \\
sk_{i,2} =& A^{\frac{-\theta_{i,1}\theta_{i,2}}{F(ID_i)}} g^{s_i} = g^{s_i - \frac{a\theta_{i,1}\theta_{i,2}}{F(ID_i)}} = g^{\tilde{s}_i},
\end{aligned}$$

where $\tilde{s}_i = s_i - \frac{a\theta_{i,1}\theta_{i,2}}{F(ID_i)}$. Hence, the above equations indicate that $sk_i = (sk_{i,1}, sk_{i,2})$ is a valid private key of identity ID_i.

– *Signing Query* $\mathcal{O}_{sign}(ID_i, T, m)$: Upon receiving a message m, an identity ID_i, and a timestamp T, $\mathcal{C}$ issues a query $\mathcal{O}_{pk}(ID_i)$ to obtain a public key $pk_i = (pk_{i,1}, pk_{i,2}, pk_{i,3})$ and a triplet $(\theta_{i,1}, \theta_{i,2}, \theta_{i,3})$. Then, $\mathcal{C}$ considers the following two cases:

(1) If $F(ID_i) \neq 0 \bmod l_u$, $\mathcal{C}$ makes a query $\mathcal{O}_{sk}(ID_i)$ to obtain a private key sk_i and then runs the algorithm **Sign** to generate a signature σ_i on m. Finally, $\mathcal{C}$ sends σ_i to $\mathcal{A}_2$.

(2) If $F(ID_i) = 0 \bmod l_u$, $\mathcal{C}$ computes $K(m)$. If $K(m) = 0 \bmod l_m$, $\mathcal{C}$ quits the simulation; otherwise, $\mathcal{C}$ randomly selects $r_i, r_m \in Z_p^*$ and computes $\vec{Q}_i = H_1(ID_i, pk_i)$, U_{ID_i}, $\vec{M} = H_2(m, T)$, and V_m. Furthermore, $\mathcal{C}$ computes

$$\begin{aligned}
\sigma_{i,2} =& g^{r_i}, \\
\sigma_{i,3} =& A^{\frac{-\theta_{i,1}\theta_{i,2}}{K(m)}} g^{r_m}, \\
h_i =& H_3(m, T, ID_i, pk_i, \sigma_{i,2}, \sigma_{i,3}, param), \\
\sigma_{i,1} =& g^{\alpha\beta} (U_{ID_i})^{r_i} A^{\frac{(L(m) - h_i\theta_{i,3})\theta_{i,1}\theta_{i,2}}{K(m)}} ((pk_{i,3})^{h_i} V_m)^{r_m}.
\end{aligned}$$

Finally, $\mathcal{C}$ sends $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, \sigma_{i,3})$ to $\mathcal{A}_2$.

Let $\tilde{r}_m = r_m - \frac{a\theta_{i,1}\theta_{i,2}}{K(m)}$; then, we have

$$\begin{aligned}
\sigma_{i,1} =& g^{\alpha\beta} (U_{ID_i})^{r_i} A^{\frac{(L(m) - h_i\theta_{i,3})\theta_{i,1}\theta_{i,2}}{K(m)}} ((pk_{i,3})^{h_i} V_m)^{r_m} \\
=& g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} (g^{bK(m)} g^{L(m)} g^{h_i\theta_{i,3}})^{\frac{-a\theta_{i,1}\theta_{i,2}}{K(m)}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{r_m} \\
=& g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} ((B^{K(m)} g^{L(m)})(g^{\theta_{i,3}})^{h_i})^{\frac{-a\theta_{i,1}\theta_{i,2}}{K(m)}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{r_m} \\
=& g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{r_m - \frac{a\theta_{i,1}\theta_{i,2}}{K(m)}} \\
=& g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{\tilde{r}_m}, \\
\sigma_{i,2} =& g^{r_i}, \\
\sigma_{i,3} =& A^{\frac{-\theta_{i,1}\theta_{i,2}}{K(m)}} g^{r_m} = g^{\frac{-a\theta_{i,1}\theta_{i,2}}{K(m)}} g^{r_m} = g^{r_m - \frac{a\theta_{i,1}\theta_{i,2}}{K(m)}} = g^{\tilde{r}_m}.
\end{aligned}$$

The simulated signature $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, \sigma_{i,3})$ satisfies the following signature verification equation; thus, σ_i is a valid signature on message m:

$$\begin{aligned} e(\sigma_{i,1}, g) =& e(g^{\alpha\beta} g^{ab\theta_{i,1}\theta_{i,2}} (U_{ID_i})^{r_i} ((pk_{i,3})^{h_i} V_m)^{\tilde{r}_m}, g) \\ =& e(g^{\alpha\beta}, g) e(g^{ab\theta_{i,1}\theta_{i,2}}, g) e((U_{ID_i})^{r_i}, g) e(((pk_{i,3})^{h_i} V_m)^{\tilde{r}_m}, g) \\ =& e(g^{\alpha}, g^{\beta}) e(g^{a\theta_{i,1}}, g^{b\theta_{i,2}}) e(U_{ID_i}, g^{r_i}) e((pk_{i,3})^{h_i} V_m, g^{\tilde{r}_m}) \\ =& e(g_2, g_1) e(pk_{i,1}, pk_{i,2}) e(U_{ID_i}, \sigma_{i,2}) e((pk_{i,3})^{h_i} V_m, \sigma_{i,3}). \end{aligned}$$

- *Forgery*: $\mathcal{A}_2$ eventually outputs a signature $\sigma^* = (\sigma_1^*, \sigma_2^*, \sigma_3^*)$ on a message m^* corresponding to an identity ID^*, a timestamp T^*, and the targeted public key pk^*. If $F(ID^*) \neq 0 \bmod p$ or $K(m^*) \neq 0 \bmod p$, $\mathcal{C}$ terminates; otherwise, $\mathcal{C}$ calculates $h^* = H_3(m^*, T^*, ID^*, pk^*, \sigma_2^*, \sigma_3^*, param)$ and then uses θ^* and $g^{\alpha\beta}$ to output the CDH value g^{ab} by calculating

$$\begin{aligned} & \frac{\sigma_1^*}{g^{\alpha\beta} (\sigma_2^*)^{J(ID^*)} (\sigma_3^*)^{L(m^*)+h^*\theta^*}} \\ =& \frac{g^{\alpha\beta} g^{ab} (U_{ID^*})^{r_{ID}^*} ((pk_3^*)^{h^*} V_{m^*})^{r_m^*}}{g^{\alpha\beta} (g^{r_{ID}^*})^{J(ID^*)} (g^{r_m^*})^{L(m^*)+h^*\theta^*}} \\ =& \frac{g^{ab} (B^{F(ID^*)} g^{J(ID^*)})^{r_{ID}^*} ((pk_3^*)^{h^*} (B^{K(m^*)} g^{L(m^*)}))^{r_m^*}}{(g^{J(ID^*)})^{r_{ID}^*} (g^{\theta^*})^{r_m^* h^*} (g^{r_m^*})^{r_m^* L(m^*)}} \\ =& g^{ab} B^{F(ID^*) r_{ID}^*} B^{K(m^*) r_m^*} \\ & \text{(since} \quad F(ID^*) = K(m^*) = 0 \bmod p) \\ =& g^{ab}. \end{aligned}$$

Here, we discuss the probability of $\mathcal{C}$ outputting a correct solution for the CDH instance. $\mathcal{C}$ completes the above simulation if all of the following events occur:

- $F(ID_i) \neq 0 \bmod l_u$ during private key queries.
- $F(ID_i) \neq 0 \bmod l_u$ or $K(m) \neq 0 \bmod l_m$ during signing queries.
- $F(ID^*) = 0 \bmod p$ and $K(m^*) = 0 \bmod p$ in the forgery phase.

The probability of $\mathcal{C}$ completing the simulation is analogous to that in Theorem 1. We define four independent events, X_i, X^*, Y_j, and Y^*, as follows:

- $X_i : F(ID_i) \neq 0 \bmod l_u$ for $1 \leq i \leq q_{sk} + q_s$.
- $X^* : F(ID^*) = 0 \bmod p$.
- $Y_j : K(m_j) \neq 0 \bmod l_m$ for $1 \leq j \leq q_s$.
- $Y^* : K(m^*) = 0 \bmod p$.

Similar to the probability analysis in Theorem 1, we give the probability of $\mathcal{C}$ not aborting as

$$\begin{aligned}\Pr[\neg abort] &= \Pr[\bigcap_{i=1}^{q_{sk}+q_s} X_i \cap X^* \cap \bigcap_{j=1}^{q_s} Y_j \cap Y^*] \\ &= \Pr[X^*]\Pr[\bigcap_{i=1}^{q_{sk}+q_s} X_i|X^*]\Pr[Y^*]\Pr[\bigcap_{j=1}^{q_s} Y_j|Y^*] \\ &= \frac{1}{l_u(n_u+1)}\left(1-\frac{q_{sk}+q_s}{l_u}\right)\frac{1}{l_m(n_m+1)}\left(1-\frac{q_s}{l_m}\right) \\ &= \frac{1}{16q_s(q_{sk}+q_s)(n_u+1)(n_m+1)}.\end{aligned}$$

Hence, $\mathcal{C}$ can solve the given instance of the CDH problem with advantage $\varepsilon_2^{'} \geq \frac{\varepsilon_2}{16q_s(q_{sk}+q_s)(n_u+1)(n_m+1)}$. □

We obtain Theorem 3 by combining Theorems 1 and 2, as follows.

Theorem 3. *In the standard model, our CLS scheme is strongly unforgeable against adaptive chosen-message attacks corresponding to type I and II adversaries under the CDH and CRHF assumptions.*

Theorem 4. *Our CLS scheme is resistant to replay attacks.*

Proof. In replay attacks, the adversary generally initiates two types of attacks [31,32]. One is to directly replay the intercepted message and the corresponding signature, and the other is to modify the timestamp in the signature of the intercepted message and to create a new signature for the message.

In the first type of attack, it is assumed that the adversary replays an intercepted combination of message m, timestamp T, and signature σ generated by an IoT device. Upon receiving this combination $\{m, T, \sigma\}$, the data centre compares the timestamp T in the combination with the current timestamp $T^{'}$. If the value of $T^{'} - T$ exceeds the threshold δ, the data centre can determine that m in this combination is a replayed message and can discard m. Therefore, the first type of attack has no effect on our CLS scheme. □

Since our CLS scheme satisfies strong unforgeability, the attacker cannot generate a legal signature for any message. Therefore, in the second type of attack, the attacker can only use the existing combination $\{m, T, \sigma\}$ to initiate the attack. In the proposed scheme, the timestamp T is bound to the message m, i.e., $\vec{M} = H_2(m, T)$. Additionally, the timestamp T is embedded in the parameter h in the form of $h = H_3(m, T, ID, pk_{ID}, \sigma_2, \sigma_3, param)$ and is also embedded in the signature σ of the message m in the form of $(pk_3)^{r_m h}$. If the attacker wants to replace T in the signature σ with a new timestamp T^*, the attacker needs to calculate $h^* = H_3(m, T^*, ID, pk_{ID}, \sigma_2, \sigma_3, param)$ and $(pk_3)^{r_m h^*}$. Although an attacker can calculate h^* and $(pk_3)^{h^*}$, the difficulty in calculating r_m from $(pk_3)^{r_m h}$ is equivalent to solving the DL problem. However, if the attacker does not know r_m, then they cannot calculate the correct value $(pk_3)^{r_m h^*}$. In addition, T^* must satisfy the conditions $H_2(m, T) = H_2(m, T^*)$ and $H_1(m, T) = H_1(m, T^*)$, which is equivalent to finding a collision of the hash functions H_2 and H_3. Since the DL problem is unsolvable in reality and the functions H_2 and H_3 are CRHF, the second type of attack does not compromise the security of our CLS scheme. In summary, the proposed CLS scheme can efficiently withstand replay attacks.

6. Application in IoT Environments and Performance Analysis

6.1. System Model

In a CLS scheme for IoT environments, it is very important that data are not modified and that the source of the data is authentic during data transmission. Therefore, we mainly focus on the integrity and authenticity of IoT data in our system while simultaneously reducing the bandwidth, computational cost, and storage overhead for IoT devices. Figure 2 shows our CLS system model for IoT environments, which consists of three entities: PKG, data centre, and IoT device.

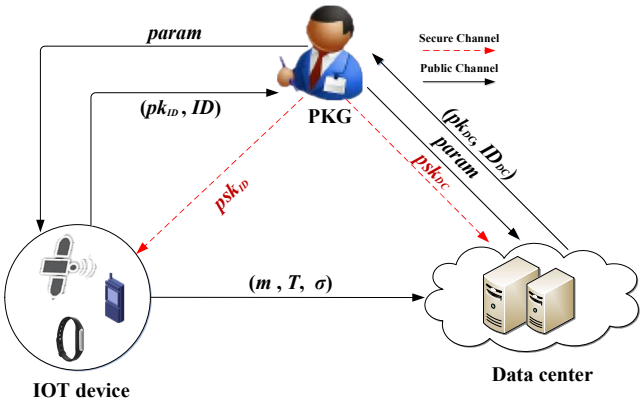

Figure 2. System model of the proposed certificateless signature (CLS) scheme for IoT.

- **PKG:** This entity is primarily responsible for producing system parameters and computing partial private keys for the data centre and each IoT device. The PKG sends system parameters to all of the entities through a public channel and transmits an individual partial private key to each entity via a secure channel.
- **Data centre:** This entity has a strong computing power and storage space; thus, it can check the integrity and authenticity of the data by verifying the signature sent by each IoT device and can store the authentic data for other users to use. Initially, the data centre submits its identity information to the PKG to apply for the corresponding partial private key; it then saves the system parameters and partial private key sent by the PKG.
- **IoT device:** This entity equipped with sensors has limited computational and memory resources and limited battery capacity. During the registration of the IoT device, the PKG generates a unique partial private key based on the physical address of each IoT device. After the IoT device is embedded with system parameters and its private key, it signs messages collected from the physical world and sends the corresponding signatures along with messages to the data centre.

6.2. Performance Analysis

In this subsection, we analyze the performance of the proposed CLS scheme. Compared with other cryptographic operations, bilinear pairing and exponentiation are the most time-consuming operations [22,28]; hence, our efficiency analysis mainly emphasizes the computational costs of these two operations. Tables 1 and 2 compare the performance of our CLS scheme and other related CLS schemes [21–23,27,30] without random oracles in terms of private key size, signature length, computational cost, and security. In Table 1, the *KeySize* and *SigSize* columns list the sizes of the private key and signature, respectively. The *Sign* and *Verify* columns present the computational costs of the algorithms **Sign** and **Verify**, respectively. Let P and E represent the execution times of a bilinear pairing and an exponentiation,

respectively. Let n_u represent the length of an identity, and let $|G_1|$ and $|p|$ represent the lengths of an element in G_1 and Z_p, respectively. In Table 2, the columns *Type I*, *Type II*, and *Replay attacks* show whether the CLS scheme can resist PKR attacks, MKGC attacks, and replay attacks, respectively. The *SUF* column denotes whether the CLS scheme satisfies a strong unforgeability in the standard model. It should be noted that the key length affects the storage capacity of the IoT device and the data center and that the signature length affects the communication capabilities of the IoT device and the storage capacity of the data center. In addition, the overhead of signature generation and signature verification affect the computing power of the IoT device and the data center, respectively.

Table 1. A comparison of the CLS scheme performance.

Scheme	KeySize	SigSize	Sign	Verify
Yu et al. [21]	$\|p\|+2\|G_1\|$	$4\|G_1\|$	$7E$	$E+5P$
Yuan et al. [23]	$\|p\|+2\|G_1\|$	$3\|G_1\|$	$3E$	$E+6P$
Pang et al. [27]	$\|p\|+2\|G_1\|$	$3\|G_1\|$	$7E$	$4E+5P$
Huang et al. [22]	$3\|G_1\|$	$3\|G_1\|$	$5E$	$3E+6P$
Yang et al. [30]	$(4+n_u)\|p\|+2\|G_1\|$	$\|p\|+4\|G_1\|$	$10E$	$3E+7P$
Our scheme	$2\|G_1\|$	$3\|G_1\|$	$3E$	$E+3P$

Table 2. A comparison of the security attributes.

Scheme	Type I	Type II	SUF	Replay Attacks
Yu et al. [21]	No	No	No	No
Yuan et al. [23]	Yes	Yes	No	No
Pang et al. [27]	Yes	Yes	No	No
Huang et al. [22]	Yes	No	No	No
Yang et al. [30]	Yes	Yes	Yes	No
Our scheme	Yes	Yes	Yes	Yes

From Tables 1 and 2, the length of the private key in our CLS scheme is $2|G_1|$, which is the shortest among the six CLS schemes. The size of the signature in the proposed CLS scheme is $3|G_1|$, which is equivalent to that of the schemes presented in References [22,23,27] but smaller than that of other schemes [21,30]. In the signing phase, our CLS scheme requires three exponentiations, as does Yuan et al.'s scheme [23], but is superior to other schemes [21,22,27,30]. In the verification phase, the computational cost of the proposed CLS scheme is $E+5P$, which is lower than that of the five other CLS schemes. Moreover, the efficiency of the verification process in our CLS scheme can be improved by a pre-calculation. Note that the verification equation for signature legitimacy is as follows:

$$e(\sigma_1, g) = e(g_2, g_1) \cdot e(pk_{ID,1}, pk_{ID,2}) \cdot e(U_{ID}, \sigma_2) \cdot e((pk_{ID,3})^h V_m, \sigma_3).$$

Here, $e(g_2, g_1)$ and $e(pk_{ID,1}, pk_{ID,2})$ can be pre-computed; thus, the time cost of verification in our CLS scheme can be reduced to one exponentiation and 3 bilinear pairings. Furthermore, only our CLS scheme can resist PKR attacks, MKGC attacks, and replay attacks while satisfying a strong unforgeability.

We also evaluated the performance of the proposed CLS scheme via experiments conducted with the PBC-0.47-VC cryptographic library [50]. The simulation program was run on a laptop equipped with a basic configuration of a 2.50 GHz CPU, 8 GB RAM, and the 64-bit Windows 10 operating system. To obtain faster pairing computation, we selected the Type A curve in the PBC library, which is a super-singular curve $y^2 = x^3 + x$ built with the 512-bit order of the base field. The results of the experiment are presented in Figures 3–6.

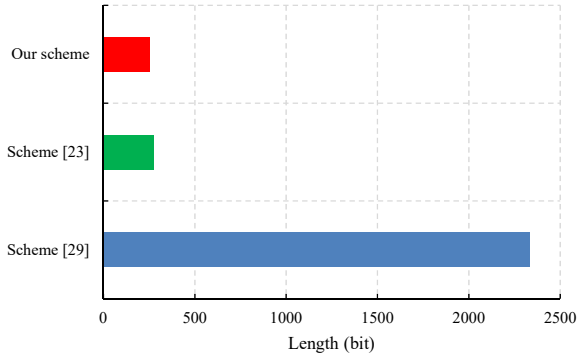

Figure 3. A comparison of the private key size.

The IoT device must secretly store its private key; therefore, the size of the private key is important for an IoT device with a limited storage capacity. As shown in Figure 3, the size of the private key in our CLS scheme is 256 bits, which is 92.8% of that in Yuan et al.'s CLS scheme [23]. However, the size of the private key increases linearly with the length of the entity's identity in Yang et al.'s CLS scheme [30]. For example, if the length n_u of the entity's identity is 100 bits, then the private key size in our CLS scheme is approximately 11% of that in Yang et al.'s CLS scheme [30]. In other words, our CLS scheme has a higher performance in private key length.

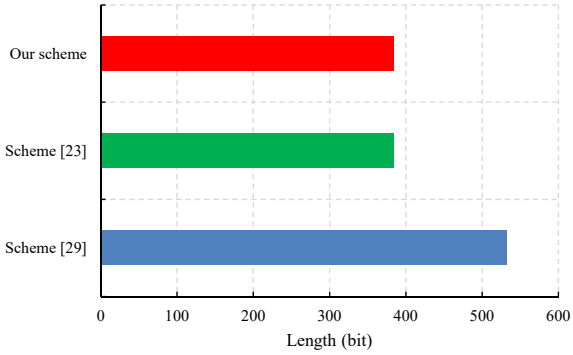

Figure 4. A comparison of the communication cost.

Since IoT devices possess limited battery power and communication bandwidth, one of the goals of our CLS scheme is to reduce the communication overhead of IoT devices. The most critical factor affecting communication cost is signature size. Figure 4 shows that the signature size of our CLS scheme and that of Yuan et al. [23] is 384 bits, while the signature size of Yang et al.'s CLS scheme [30] is 532 bits. Hence, the proposed CLS scheme has a lower communication overhead.

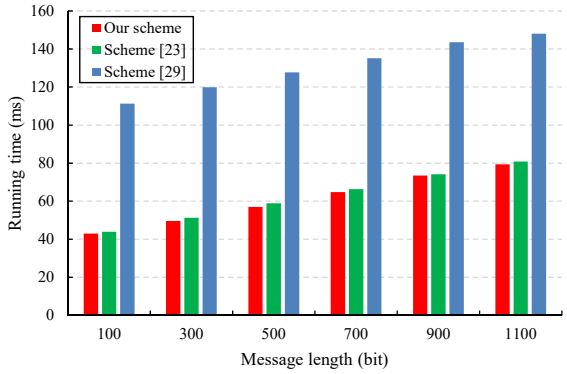

Figure 5. A comparison of the signature generation cost.

Due to the characteristics of IoT devices, such as limited computing and processing power, the computational overhead of generating signatures for IoT devices should be as small as possible. Figure 5 shows that the cost of signature generation in our CLS scheme is almost the same as that in Yuan et al.'s CLS scheme [23] but less than that in Yang et al.'s CLS scheme [30].

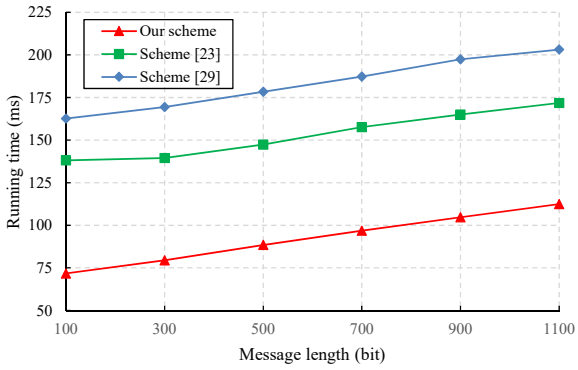

Figure 6. A comparison of the signature verification cost.

The data centre has a strong computation and storage capability to verify the validity of signatures sent by IoT devices. Figure 6 shows that the proposed CLS scheme greatly reduces the computational overhead of signature verification and that its performance is superior to that of the other two schemes [23,30].

A scheme in the random oracle model usually has a higher computational performance, but its security depends on the ideal random oracle. Both our scheme and Yang et al.'s [48] scheme are provable in the standard model, and their security only depends on the difficulty of the associated mathematical problems. Therefore, these two schemes have higher security than other schemes [4,40–42,44,46,47]. Our scheme and Yang et al.'s scheme [48] use CLS and SDVPRS respectively to guarantee the integrity and authenticity of data in IoT. We compare the signature generation and verification overhead of two schemes, and the corresponding results are shown in Figures 7 and 8.

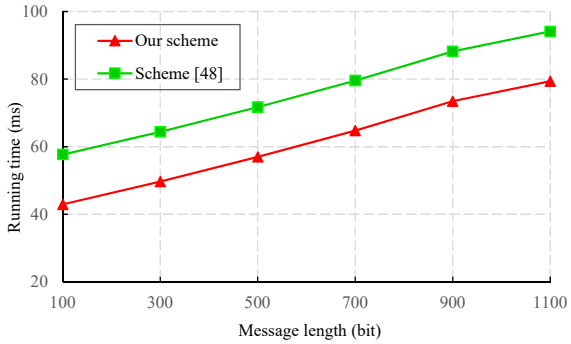

Figure 7. A comparison of the signature generation cost between CLS-based and SDVPRS-based authentication schemes.

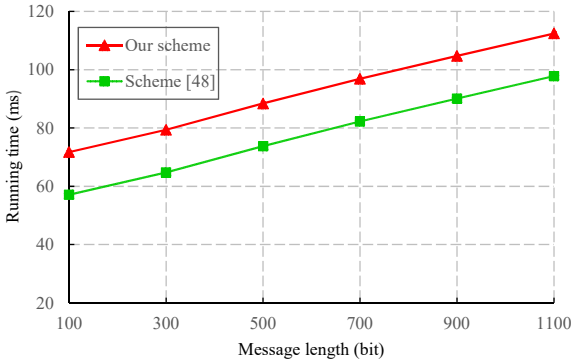

Figure 8. A comparison of the signature verification cost between CLS-based and SDVPRS-based authentication schemes.

From Figure 7, we can see that the computational cost of signature generation in our scheme is lower than that in the scheme of Reference [48]. This is because the signature generation in the scheme of Reference [48] requires an additional bilinear pairing operation. Figure 8 shows that the time consumption of signature verification in the scheme of Reference [48] is lower than ours, but the scheme in Reference [48] does not satisfy the properties of a strong unforgeability and replay attack resistance. As a result, our scheme has a higher security.

In summary, the results of all the above experimental analyses are consistent with those of the theoretical analysis in Table 1. Therefore, we conclude that our CLS scheme is applicable to IoT environments.

7. Conclusions

The IoT is profoundly changing production activities, social management, and public services, but ensuring the integrity and authenticity of data is an important issue for IoT. To solve this problem, a new CLS scheme for IoT environments is presented in this paper. In addition to protecting data integrity and data authenticity, our CLS scheme also reduces the computational and communication costs for IoT devices. The proposed CLS scheme is proven to be strongly unforgeable against adaptive chosen-message attacks under the CDH and CRHF assumptions in the standard model. Additionally, our CLS scheme can withstand replay attacks. Furthermore, the performance comparisons demonstrate that our CLS scheme outperforms the previous CLS schemes without random oracles. The Internet of Vehicles is considered

to be one of the most potential areas in IoT and has wide application prospects in the field of intelligent transportation. Compared with ordinary sensors, the vehicle terminal equipment has a more stable power supply and higher computing power and storage space. Hence, our CLS scheme is suitable for the Internet of Vehicles.

Author Contributions: X.Y. wrote the original draft. X.P. conducted security analysis. G.C. and T.L. designed the simulation experiments. M.W. re-edited the draft. C.W. validated the rightness and feasibility of the proposed scheme.

Funding: This research was funded by the National Natural Science Foundation of China under Grant 61662069, the China Postdoctoral Science Foundation under Grant 2017M610817, and the Foundation for Excellent Young Teachers by Northwest Normal University under Grant NWNU-LKQN-14-7.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, Y.; Wu, L.; Yin, G.; Li, L.; Zhao, H. A survey on security and privacy issues in Internet-of-Things. *IEEE Internet Things J*. **2017**, *4*, 1250–1258. [CrossRef]
2. Shen, L.; Ma, J.; Liu, X.; Wei, F.; Miao, M. A secure and efficient id-based aggregate signature scheme for wireless sensor networks. *IEEE Internet Things J*. **2017**, *4*, 546–554. [CrossRef]
3. Karati, A.; Islam, S.H.; Karuppiah, M. Provably secure and lightweight certificateless signature scheme for IIoT environments. *IEEE Trans. Ind. Inform*. **2018**, *14*, 3701–3711. [CrossRef]
4. Yeh, K.H.; Su, C.; Choo, K.K.R.; Chiu, W. A novel certificateless signature scheme for smart objects in the Internet-of-Things. *Sensors* **2017**, *17*, 1001. [CrossRef] [PubMed]
5. Conti, M.; Dehghantanha, A.; Franke, K.; Watson, S. Internet of Things security and forensics: Challenges and opportunities. *Future Gener. Comput. Syst*. **2018**, *78*, 544–546. [CrossRef]
6. Perlman, R. An overview of PKI trust models. *IEEE Netw*. **2018**, *13*, 38–43. [CrossRef]
7. Shamir, A. Identity-based cryptosystems and signature schemes. In Proceedings of the CRYPTO 1984, Santa Barbara, CA, USA, 19–22 August 1984; pp. 47–53.
8. Al-Riyami, S.S.; Paterson, K.G. Certificateless public key cryptography. In Proceedings of the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, 30 November–4 December 2003; pp. 452–473.
9. Yum, D.H.; Lee, P.J. Generic construction of certificateless signature. In Proceedings of the Australasian Conference on Information Security and Privacy, Sydney, Australia, 13–15 July 2004; pp. 200–211.
10. Wan, Z.; Weng, J.; Li, J. Security mediated certificateless signatures without pairing. *J. Comput*. **2010**, *12*, 1862–1869. [CrossRef]
11. Xiong, H.; Guan, Z.; Chen, Z.; Li, F. An efficient certificateless aggregate signature with constant pairing computations. *Inf. Sci*. **2013**, *219*, 225–235. [CrossRef]
12. He, D.; Tian, M.; Chen, J. Insecurity of an efficient certificateless aggregate signature with constant pairing computations. *Inf. Sci*. **2014**, *268*, 458–462. [CrossRef]
13. Chen, Y.C.; Tso, R.; Mambo, M.; Huang, K.; Horng, G. Certificateless aggregate signature with efficient verification. *Secur. Commun. Netw*. **2015**, *13*, 2232–2243. [CrossRef]
14. Kang, B.; Wang, M.; Jing, D. An efficient certificateless aggregate signature scheme. *Wuhan Univ. J. Nat. Sci*. **2017**, *22*, 165–170. [CrossRef]
15. Wang, L.; Chen, K.; Long, Y.; Wang, H. An efficient pairing-free certificateless signature scheme for resource-limited systems. *Sci. China Inf. Sci*. **2017**, *60*, 119102. [CrossRef]
16. Bellare, M.; Rogaway, P. The exact security of digital signatures-How to sign with RSA and Rabin. In Proceedings of the Theory and Applications of Cryptographic Techniques, Konstanz, Germany, 2–6 May 1999; pp. 399–416.
17. Canetti, R.; Goldreich, O.; Halevi, S. The random oracle methodology, revisited. *J. ACM* **2004**, *51*, 557–594. [CrossRef]

18. Liu, J.K.; Au, M.H.; Susilo, W. Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model. In Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security, Singapore, 20–22 March 2007; pp. 273–283.
19. Xiong, H.; Qin, Z.; Li, F. An improved certificateless signature scheme secure in the standard model. *Fundam. Inf.* **2008**, *88*, 193–206.
20. Yuan, Y.; Li, D.; Tian, L.; Zhu, H. Certificateless signature scheme without random oracles. In Proceedings of the Information Security and Assurance, Seoul, Korea, 18–20 August 2009; pp. 31–40.
21. Yu, Y.; Mu, Y.; Wang, G.; Xia, Q.; Yang, B. Improved certificateless signature scheme provably secure in the standard model. *IET Inf. Secur.* **2012**, *6*, 102–110. [CrossRef]
22. Hung, Y.H.; Huang, S.S.; Tseng, Y.M.; Tsai, T.T. Certificateless signature with strong unforgeability in the standard model. *Informatica* **2016**, *26*, 663–684. [CrossRef]
23. Yuan, Y.; Wang, C. Certificateless signature scheme with security enhanced in the standard model. *Inf. Process. Lett.* **2014**, *114*, 492–499. [CrossRef]
24. Tsai, T.T.; Huang, S.S.; Tseng, Y.M. Secure certificateless signature with revocation in the standard model. *Math. Probl. Eng.* **2014**, *2014*, 1–16. [CrossRef]
25. Canard, S.; Trinh, V.C. An Efficient certificateless signature scheme in the standard model. In Proceedings of the Information Systems Security, Rome, Italy, 8–9 December 2016; pp. 175–192.
26. Waters, B. Efficient identity-based encryption without random oracles. In Proceedings of the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005; pp. 114–127.
27. Pang, L.; Zhao, H.; Zhou, X.; Li, H. Strongly unforgeable and efficient proxy signature scheme with fast revocation secure in the standard model. *Int. J. Distrib. Sens. Netw.* **2016**, *12*, 1–12. [CrossRef]
28. Tsai, T.T.; Tseng, Y.M.; Huang, S.S. Efficient strongly unforgeable ID-based signature without random oracles. *Informatica* **2014**, *25*, 505–521. [CrossRef]
29. Kwon, S. An identity-based strongly unforgeable signature without random oracles from bilinear pairings. *Inf. Sci.* **2014**, *276*, 1–9. [CrossRef]
30. Yang, W.; Weng, J.; Luo, W.; Yang, A. Strongly Unforgeable Certificateless Signature Resisting Attacks from Malicious-But-Passive KGC. *Secur. Commun. Netw.* **2017**, *5704865*, 1–8. [CrossRef]
31. Huang, Y.; Zhang, X.; Yu, B. Efficient anti-replay identity-based signature scheme for wireless body area network. *J. Cryptol. Res.* **2017**, *4*, 447–457.
32. Pei, H.L.; Shang, T.; Liu, J.W. Secure network coding method merged with timestamp and homomorphic signature. *J. China Inst. Commun.* **2013**, *34*, 28–35.
33. Huang, X.; Susilo, W.; Mu, Y.; Zhang, F. On the security of certificateless signature schemes from Asiacrypt 2003. In Proceedings of the Cryptology and Network Security, Xiamen, China, 14–16 December 2005; pp. 13–25.
34. Paterson, K.G.; Schuldt, J.C. Efficient identity-based signatures secure in the standard model. In Proceedings of the Australasian Conference on Information Security and Privacy, Melbourne, Australia, 3–5 July 2006; pp. 207–222.
35. Huang, X.; Mu, Y.; Susilo, W.; Wong, D.S.; Wu, W. Certificateless signature revisited. In Proceedings of the Australasian Conference on Information Security and Privacy, Townsville, Australia, 2–4 July 2007; pp. 308–322.
36. Shim, K A.; Lee, Y.R. Security pitfalls of the certificateless signature and multi-receiver signcryption schemes. *Fund. Inf.* **2011**, *112*, 365–376.
37. Xia, Q.; Xu, C.X.; Yu, Y. Key replacement attack on two certificateless signature schemes without random oracles. *Key Eng. Mater.* **2010**, *439*, 1606–1611. [CrossRef]
38. Boneh, D.; Boyen, X. Short signatures without random oracles. In Proceedings of the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, 13–17 April 2008; pp. 56–73.
39. Pointcheval, D.; Sanders, O. Short randomizable signatures. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 29 February–4 March 2016; pp. 111–126.
40. Jia, X.; He, D.; Liu, Q.; Choo, K.K.R. An efficient provably-secure certificateless signature scheme for Internet-of-Things deployment. *Ad Hoc Netw.* **2018**, *71*, 78–87. [CrossRef]

41. Li, X.; Wang, H.; Yu, Y.; Qian, C. An IoT data communication framework for authenticity and integrity. In Proceedings of the IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation, Pittsburgh, PA, USA, 18–21 April 2017; pp. 159–170.
42. Frädrich, C.; Pöhls, H.C.; Popp, W.; Rakotondravony, N.; Samelin, K. Integrity and authenticity protection with selective disclosure control in the cloud and IoT. In Proceedings of the International Conference on Information and Communications Security, Singapore, 29 November–2 December 2016; pp. 197–213.
43. Steinfeld, R.; Bull, L.; Zheng, Y. Content extraction signatures. In Proceedings of the International Conference on Information Security and Cryptology, Seoul, Korea, 6–7 December 2001; pp. 285–304.
44. Challa, S.; Wazid, M.; Das, A.K.; Kumar, N.; Reddy, A.G.; Yoon, E.J.; Yoo, K.Y. Secure signature-based authenticated key establishment scheme for future IoT applications. *IEEE Access* **2017**, *5*, 3028–3043. [CrossRef]
45. Nyberg, K. Fast accumulated hashing. In Proceedings of the International Workshop on Fast Software Encryption, Cambridge, UK, 21–23 February 1996; pp. 83–87.
46. Yao, X.; Han, X.; Du, X.; Zhou, X. A lightweight multicast authentication mechanism for small scale IoT applications. *IEEE Sens. J.* **2013**, *13*, 3693–3701. [CrossRef]
47. Yang, X.; Chen, C.; Ma, T.; Li, Y.; Wang, C. An improved certificateless aggregate signature scheme for vehicular ad-hoc networks. In Proceedings of the IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference, Chongqing, China, 12–14 October 2018; pp. 2334–2338.
48. Yang, X.D.; Xiao, L.K.; Chen, C.L.; Wang, C.F. A strong designated verifier proxy re-signature scheme for IoT environments. *Symmetry* **2018**, *10*, 580. [CrossRef]
49. Au, M.H.; Mu, Y.; Chen, J.; Wong, D.S.; Liu, J.K.; Yang, G. Malicious KGC attacks in certificateless cryptography. In Proceedings of the 2nd ACM symposium on Information, Computer and Communications Security, Singapore, 20–22 March 2007; pp. 302–311.
50. Lynn, B. The Pairing-Based Cryptography Library. Available online http://crypto.stanford.edu/pbc (accessed on 14 June 2019).