IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 17, 2019, accepted August 3, 2019, date of publication August 14, 2019, date of current version August 27, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2935252

Top-Level Secure Certificateless Signature Against Malicious-But-Passive KGC

WENJIE YANG, SHANGPENG WANG, WEI WU, AND YI MU
Fujian Provincial Key Laboratory of Network Security and Cryptology, Fujian Normal University, Fuzhou 350007, China
College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350117, China

Corresponding author: Wei Wu (weiwu@fjnu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61822202, Grant 61872087, and Grant 61872089.

ABSTRACT Certificateless signature (CLS) has no need of public key certificates and also avoids excessive dependence to a third party like that in identity-based setting. Recently, Shim (IEEE Systems Journal, doi:10.1109/JSYST.2018.2844809) came up with a CLS scheme independent of random oracles and asserted that the construction can be immune to the public key replacement attacks and the malicious-but-passive key generation center (KGC) attacks. In this paper, we analyze the security of Shim's scheme and point out that his conclusions are incorrect by giving two concrete counter-examples. We repair the scheme and put forward a CLS scheme secure against public key replacement attacks and malicious-but-passive KGC attacks without relying on random oracles. Compared with Shim's scheme, our construction has lower execution cost for signing and verification, and achieves Girault's top-level security, which means that a victim can repudiate the forgeries based on a false secret key generated by the KGC.

INDEX TERMS Malicious-but-passive KGC attacks, public key replacement attacks, certificateless signature, top-level security, bilinear pairings, standard model.

I. INTRODUCTION

Digital signatures can assure the validity, completeness, and non-repudiation of data resources and have drawn a lot of interest since their introduction. In deployment, however, Certification Authority (CA) needs to be deployed to guarantee the relationship between a verification key and its holder, and any verifier needs to check the verification key validity before trusting a digital signature, which is tedious, time consuming, and inefficient.

In 1984, Shamir [1] conceived the identity-based cryptography (IBC). In such a scenario, the acknowledged entity identity is directly considered as its public key and the corresponding private key can be derived from the identity by a private key generator (PKG). Here, the cumbersome certification like that from CA has been avoided. For another, PKG can impersonate any entity owning to know all entities private key. Obviously, key escrow is inevitably brought into IBC.

In 2003, Al-Riyami and Paterson [2] put forth a primitive of certificateless signature (CLS) to overcome these weaknesses in the previous cryptosystems. In CLS, each entity not only independently chooses his/her secret value but also requests a partial private key from a key generation center (KGC) to initialize the full secret key for themselves. Clearly, the secret value and the partial private key make up of the entity full secret signing key, which are generated by two independent parties. Only holding one of the above two parts cannot affect system security. In other words, neither KGC who just knows a target entity partial private key nor any interested party who just updates an uncertified target entity public key can generate a valid signature for the target entity. However, most previous studies depend on random oracle model (ROM) [3]. Unfortunately, when using concrete hash functions substitutes ideal ROMs, these studies are no longer guaranteed security in realty. The CLS schemes without ROM are more attractive.

In [4], Shim presented an efficient CLS scheme and declared that its security can be ensured without depending on random oracles. Nevertheless, in this paper, we find that Shim's scheme cannot resist these attacks launched by the public key replacement attacker and the malicious-but-passive KGC, and gave two concrete attacks to illustrate that the security argument showed in [4] fails. We also put forth an efficient construction and prove its security against

The associate editor coordinating the review of this article and approving it for publication was Zhitao Guan.

public key replacement attacks and malicious-but-passive KGC attacks without using random oracles. Compared with Shim's scheme, our construction has lower execution cost for signing and verification, and achieves Girault's top-level security, which means that a victim is able to repudiate the forgeries based on a false key pair produced by KGC. Note that, the details of Girault's security level is concisely reviewed in Subsection II-B.

A. RELATED WORK

Certificateless signature (CLS) [2] was first introduced by Al-Riyami and Paterson in Asiacrypt'03. Here, the key generation center (KGC) only produces a user's partial private key, and each user picks an additional secret value for themselves independently. Obviously, the certificates management and key escrow problems in traditional public key system and identity-based system respectively are overcame in CLS. Unfortunately, Huang *et al.* [5] indicated that the concrete scheme given in [2] cannot resist the public key replacement attack. Meanwhile, they formally defined the security model of CLS and proposed an improvement. Later, a lot of useful schemes [6]–[18] were introduced to optimize performance. Nevertheless, most early studies were only proven secure in random oracle model, and some did not even provide rigorous proofs, whose security has no theoretical foundation.

In 2007, Liu *et al.* [19] raised the first CLS scheme provably secure in the standard model (without ROM). Nevertheless, Xiong *et al.* [9] pointed out that Liu et al.'s scheme cannot withstand the malicious-but-passive KGC attacks and gave a new construction. The next year, Yuan *et al.* [20] introduced another CLS scheme and claimed that it can be proven secure in the standard model. Unfortunately, two concrete public key replacement attacks on both of them [9], [20] were illustrated by Xia *et al.* [21]. Later, Yu *et al.* [22] proposed an improved CLS scheme with higher computational efficiency and shorter system parameters without ROM. In 2014, Yuan and Wang [23] illustrate that Yu et al.'s CLS scheme is still subjected to the attacks from public key replacement adversaries and malicious-but-passive KGC, and then gave a resultful modification. In 2015, Pang *et al.* [24] constructed a new CLS scheme and asserted that the new scheme can reach Girault's trust level 3 in the standard model. In 2017, Wang and Xu [25] showed that [24] still cannot resist the malicious-but-passive KGC attacks and proposed a new construction in the standard model. In [25], the signature size is related to the output length of hush functions, which is not very practical. After that, a strongly unforgeable CLS scheme was given in [26] but it can meet Girault's trust level 3. In [27], Tseng *et al.* made a summary for the existing typical CLS schemes [9]–[12], [14], [15], [22], [23], [25], [26], [28], [29] and introduced a top-level secure CLS scheme with the current optimal performance in the standard model. Almost at the same time, Shim [4] gave a more efficient CLS scheme without using random oracles. Unfortunately, we will demonstrate that Shim's CLS scheme is still vulnerable to the key replacement attacks and the malicious-but-passive KGC attacks.

The paper has the following organization. Some preliminaries are given in Section II. Then, Section III shows a cryptanalysis on Shim's security argument. Next, our CLS construction and its security proof are introduced in Section IV. In Section V, we make a comparison with Shim's scheme. The overview is provided in Section VI.

II. PRELIMINARIES

A. BILINEAR GROUPS AND DIFFICULTY ASSUMPTIONS

Bilinear groups The bilinear map is defined as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ denote two q-order multiplicative cyclic groups. It has the following features:

•Bilinearity: $\hat{e}(u_1^x, u_2^y) = \hat{e}(u_1, u_2)^{xy}$, where $\forall u_1, u_2 \in \mathbb{G}_1$ and $\forall x, y \in \mathbb{Z}_p^*$;

•Non-degenracy: $\hat{e}(g, g) \neq 1_{\mathbb{G}_2}$, where g and $1_{\mathbb{G}_2}$ denote the generator of $\mathbb{G}_1$ and the identity element of $\mathbb{G}_2$, respectively;

•Computability: Calculating $\hat{e}(v_1, v_2)$ is feasible in polynomial time, where v_1, v_2 are chosen randomly from $\mathbb{G}_1$.

Throughout this paper, $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, g)$ denotes an instance above, which has the same definition in [5], [27].

Discrete Logarithm (DL) Assumption In consideration of $\langle \mathbb{G}_1, g, g' \rangle$, no polynomial time algorithm can find the integer θ from $\mathbb{Z}_p^*$ such that $g' = g^{\theta}$.

Computational Diffie-Hellman (CDH) Assumption Based on $\langle \mathbb{G}_1, g, g^{\mu}, g^{\nu} \rangle$, no polynomial time algorithm can find the group element h such that $h = g^{\mu\nu}$.

Collision Resistant Hash (CRH) Assumption Taking as input a hashing $H_k : \{0, 1\}^* \rightarrow \{0, 1\}^n$, no polynomial time algorithm can find two random values m_1, m_2 from $\mathbb{Z}_p^*$ such that $H_k(m_0) = H_k(m_1)$].

B. KGC'S SECURITY LEVEL

In [30], Girault divided the trust hierarchy to an authoritative third party into three levels. The higher the level, the lesser dependent users become on the third party. Similarly, there are also three trust hierarchies to the KGC in certificateless signatures, which are shortly revisited as follows.

- Level 1: KGC is able to obtain any legitimate entity secret key. Namely, the KGC can sign any message picked by himself/herself instead of any entity.
- Level 2: KGC can provide a false secret key for any valid entity and the victim is not able to repudiate the forging process.
- Level 3: KGC cannot replace any legitimate entity secret key with a false secret key on condition that the replacement is not noticed by the victim.

A CLS scheme achieving Level-3 security means that the KGC in the scheme does not impersonate any user by generating his/her false secret key without being detected by the victim. More specifically, the KGC cannot provide the same partial private key for different public keys.

C. OUTLINE OF CLS AND ITS SECURITY MODEL

The following five polynomial algorithms compose the generic construction of certificateless signature:

- **Setup.** Inputting a security parameters 1^λ, the KGC executes this algorithm and sets the master secret key msk and the corresponding public parameters pp.
- **UserKeyGenerating.** Inputting pp, a user with an identity ID runs this algorithm and sets the secret value e_{ID} and the corresponding public key pk_{ID}. Note that, e_{ID} is kept secret for all, including KGC.
- **PartialPrivateKeyExtracting.** Inputting pp, ID, pk_{ID} and msk, KGC runs this algorithm, and then sets and secretly transmits d_{ID} to the user as his/her partial private key. Note that, $sk_{ID} = (e_{ID}, d_{ID})$.
- **Signing.** Inputting pp, ID, pk_{ID}, sk_{ID} and a message m, the user runs this algorithm and sets a signature σ for himself/herself.
- **Verifying.** Inputting pp, ID, pk_{ID}, m and σ, a verifier runs this algorithm and returns either "TRUE" or "FALSE" in terms of the validity of σ.

Here, we also take into account three categories of attackers like in [27]. The first category denotes a public key replacement attacker ($\mathcal{A}_1$, for short) and requires that the attacker cannot know a victim partial private key but can independently update the victim secret value. The second category denotes a malicious-but-passive KGC ($\mathcal{A}_2$, for short) and requires that the attacker cannot obtain the secret value picked by a victim himself/herself but can adaptively initialize the system parameters. The third category denotes the Level 3 attacker ($\mathcal{A}_3$, for short) defined in the subsection above. Next, in order to capture all of them, we formalize the following three simulation games between a challenger $\mathcal{C}$ and $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$, respectively.

Game 1 (for the first category $\mathcal{A}_1$)

- **Init:** Inputting a security parameter 1^λ, the challenger $\mathcal{C}$ simulates **Setup** to initialize the public parameters pp and the master secret key msk. Note that the attacker $\mathcal{A}_1$ just eventually obtains pp.
- **Queries:** During this period, the attacker $\mathcal{A}_1$ can adaptively launch some queries as follows:
  $\mathcal{O}^{pk}(ID)$: Inputting an identity ID, the challenger $\mathcal{C}$ simulates **UserKeyGenerating** to generate the corresponding public key pk_{ID} for the attacker $\mathcal{A}_1$.
  $\mathcal{O}^{rep}(ID, pk'_{ID})$: The challenger $\mathcal{C}$ updates the original public key for the identity ID with the new value pk'_{ID} provided by the attacker $\mathcal{A}_1$.
  $\mathcal{O}^{ppk}(ID, pk_{ID})$: Inputting an identity ID and its public key pk_{ID}, the challenger $\mathcal{C}$ simulates **PartialPrivateKeyExtracting** to generate the corresponding partial private key d_{ID} for the attacker $\mathcal{A}_1$.
  $\mathcal{O}^{sv}(ID)$: Inputting an identity ID, the challenger $\mathcal{C}$ simulates **UserKeyGenerating** to generate the corresponding secret value e_{ID} for the attacker $\mathcal{A}_1$. Here, if the attacker $\mathcal{A}_1$ has already queried $\mathcal{O}^{rep}(ID, pk'_{ID})$ on the target identity ID, the challenger C cannot provide the corresponding secret value.
  $\mathcal{O}^{sign}(ID, pk_{ID}, m)$. Inputting a message m, an identity ID and its public key pk_{ID}, the challenger $\mathcal{C}$ simulates **PartialPrivateKeyExtracting** and **UserKeyGenerating** to obtain sk_{ID} and then performs **Signing** to generate the signature σ on m under ID and pk_{ID} for the attacker $\mathcal{A}_1$.
- **Forgery:** The attacker $\mathcal{A}_1$ makes a successful attack if he/she can give a valid forgery σ^* on (ID^*, pk_{ID^*}, m^*) such that
  (a) The item (ID^*, pk_{ID^*}) has not been taken as input in $\mathcal{O}^{ppk}(ID, pk_{ID})$;
  (b) The item (ID^*, pk_{ID^*}, m^*) has not been taken as input in $\mathcal{O}^{sign}(ID, pk_{ID}, m)$.

Game 2 (for the second category $\mathcal{A}_2$)

- **Setup:** The attacker $\mathcal{A}_2$ adaptively simulates the system parameters (pp, msk) and sends them to the challenger $\mathcal{C}$. Here, the distribution of the above parameters is indistinguishable from that of real system parameters.
- **Queries:** During this period, $\mathcal{O}^{pk}(ID)$, $\mathcal{O}^{rep}(ID, pk'_{ID})$, $\mathcal{O}^{ppk}(ID, pk_{ID})$, $\mathcal{O}^{sv}(ID)$, and $\mathcal{O}^{sign}(ID, pk_{ID}, m)$ are formalized by the challenger $\mathcal{C}$ like in **Game 1** and the attacker $\mathcal{A}_2$ can adaptively query them. Note that if the attacker $\mathcal{A}_2$ has updates the identity public key pk_{ID}, then the challenger $\mathcal{C}$ cannot return the corresponding secret value or a valid signature under the identity ID with the new public key pk'_{ID}.
- **Forgery:** The attacker $\mathcal{A}_2$ makes a successful attack if he/she can give a valid forgery σ^* on (ID^*, pk_{ID^*}, m^*) such that:
  (a) The items ID^* and (ID^*, pk_{ID^*}) have not been taken as input in $\mathcal{O}^{sv}(ID)$ and $\mathcal{O}^{rep}(ID, pk'_{ID})$, respectively;
  (b) The item (ID^*, pk_{ID^*}, m^*) has not been taken as input in $\mathcal{O}^{sign}(ID, pk_{ID}, m)$.

Definition 1 (Existential Unforgeability): Certificateless signature satisfies existential unforgeability (EUF) if any efficient attacker is unable to break the above two simulation games in a probabilistic polynomial time (PPT).

Game 3 (for the third category $\mathcal{A}_3$)

- **Setup:** The attacker $\mathcal{A}_3$ adaptively simulates the system parameters (pp, msk) and sends them to the challenger $\mathcal{C}$. Here, the distribution of the above parameters is indistinguishable from that of real system parameters.
- **Queries:** During this period, $\mathcal{O}^{pk}(ID)$, $\mathcal{O}^{rep}(ID, pk'_{ID})$, $\mathcal{O}^{ppk}(ID, pk_{ID})$, $\mathcal{O}^{sv}(ID)$, and $\mathcal{O}^{sign}(ID, pk_{ID}, m)$ are formalized by the challenger $\mathcal{C}$ like in **Game 1** and the attacker $\mathcal{A}_3$ can adaptively query them. Note that if the attacker $\mathcal{A}_3$ has updates the identity public key pk_{ID}, then the challenger $\mathcal{C}$ cannot return the corresponding secret value or a valid signature under the identity ID with the new value pk'_{ID}.
- **Forgery:** The attacker $\mathcal{A}_3$ makes a successful attack if he/she can give a valid key pair (pk_{ID^*}, sk_{ID^*}) for the target identity ID^* such that:

(a) ID^* has requested $\mathcal{O}^{pk}(ID)$ and $\mathcal{O}^{ppk}(ID, pk_{ID})$;
(b) pk_{ID^*} is not from $\mathcal{O}^{pk}(ID)$ and $\mathcal{O}^{rep}(ID, pk'_{ID})$.

Definition 2 (Top Level Security): The signature from certificateless settings satisfies the level 3 security defined in II.B, if all PPT attackers cannot break the simulation game 3 above.

III. ANALYSIS OF SHIM'S SCHEME

A. REVIEW ON SHIM'S CONSTRUCTION

Here, the five algorithms of Shim's scheme [4] are concisely revisited as follows:

- **Setup.** On the basis of $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, g)$, KGC:
  - picks randomly $\alpha \in \mathbb{Z}_p^*$, $g_2, g_3 \in \mathbb{G}_1$ and calculates $g_1 = g^\alpha$ and $Z = \hat{e}(g_1, g_2)$, and then sets $msk = g_2^\alpha$ as the master secret key.
  - selects two concrete cryptographic hashing $H_d : \{0,1\}^* \rightarrow \{0,1\}^{n_d}$, and $H_e : \{0,1\}^* \rightarrow \{0,1\}^{n_e}$, where n_d and n_e are fixed lengths.
  - chooses $d', d_1, d_2, \ldots, d_{n_d}, e', e_1, e_2, \ldots, e_{n_e} \in_R \mathbb{G}_1$, and sets $\vec{d} = \{d_i\}_{i=1}^{n_d}$, $\vec{e} = \{e_i\}_{i=1}^{n_e}$.
  - publishes $pp = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, g, g_1, g_2, g_3, d', \vec{d}, e', \vec{e}, Z)$ as the public parameters and keeps $msk = g_2^\alpha$ private.
- **UserKeyGenerating (UKG).** Inputting an identity ID, the user:
  - chooses randomly $\tau, x \in \mathbb{Z}_p^*$ and sets
  $$v_{ID} = (v_1, v_2) = (\tau, x)$$
  as the identity secret value.
  - computes and sets
  $$pk_{ID} = (pk_1, pk_2) = (g^\tau, g^x)$$
  as the identity public key.
- **PartialPrivateKeyExtracting (PPKE).** Inputting an identity ID, KGC:
  - calculates $d = H_d(ID)$, and sets $\mathcal{D} = \{i | d[i] = 1\}$, where $d[i]$ denotes the ith bit of d.
  - picks $r_d \in_R \mathbb{Z}_p^*$ and calculates
  $$\begin{aligned} s_{ID} &= (s_1, s_2) \\ &= (g_2^\alpha (D)^{r_d}, g^{r_d}), \end{aligned}$$
  where $D = d' \prod_{i \in \mathcal{D}} d_i$.
  - transmits securely the partial private key s_{ID} to the identity ID. Note that the identity full secret key is initialized to $sk_{ID} = (s_{ID}, v_{ID})$.
- **Signing.** Taking pp and a message m, the user:
  - parses sk_{ID} as (s_1, s_2, v_1, v_2) and calculates $e = H_e(m, ID, pk_{ID})$.
  - sets $\mathcal{E} = \{i | e[i] = 1\}$, where $e[i]$ stands for the ith bit of e.
  - chooses randomly $r, k \in \mathbb{Z}_p^*$ and calculates
  $$\begin{aligned} \sigma &= (\sigma_1, \sigma_2, \sigma_3) \\ &= ((s_1 \cdot D^r \cdot g_3^{v_1} \cdot E^k)^{v_2^{-1}}, s_2 \cdot g^r, g^k) \\ &= ((g_2^\alpha \cdot D^{r_d + r} \cdot g_3^x \cdot E^k)^{\tau^{-1}}, g^{r_d + r}, g^k), \end{aligned}$$
  where $D = d' \prod_{i \in \mathcal{D}} d_i$, $E = e' \prod_{i \in \mathcal{E}} e_i$.
- **Verifying.** Given pp, pk_{ID}, m, σ, a verifier:
  - computes $d = H_d(ID)$, $e = H_e(m, ID, pk_{ID})$.
  - sets $\mathcal{D} = \{i | d[i] = 1\}$ and $\mathcal{E} = \{i | e[i] = 1\}$, where $d[i]$ and $e[i]$ stand for the ith bit of d and e, respectively.
  - checks the following equation:
  $$\hat{e}(\sigma_1, pk_1) \stackrel{?}{=} Z \cdot \hat{e}(pk_2, g_3) \cdot \hat{e}(D, \sigma_2) \cdot \hat{e}(E, \sigma_3).$$
  where $D = d' \prod_{i \in \mathcal{D}} d_i$, $E = e' \prod_{i \in \mathcal{E}} e_i$.
  - outputs "TRUE" if the above formula holds; otherwise, outputs "FALSE".

B. SECURITY ANALYSIS TO SHIM'S SCHEME

1) PUBLIC KEY REPLACEMENT ATTACKS

Here, we illustrate that an attacker $\mathcal{A}_1$ who does not obtain the master secret key or the target identity partial private key, can generate a valid forgery σ^* on any message m^* under the false public key $pk'_{ID^*} = (pk'_1, pk'_2)$ picked by $\mathcal{A}_1$ for the target identity ID^*.

Stage 1. The challenger $\mathcal{C}$ normally runs the **Setup** algorithm to produce $msk = g_2^\alpha$ and $pp = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, g, g_1, g_2, g_3, d', \vec{d}, e', \vec{e}, Z)$. Here, $\mathcal{C}$ transmits pp to $\mathcal{A}_1$ and makes msk private.

Stage 2. The attacker $\mathcal{A}_1$ chooses randomly x, y from $\mathbb{Z}_p^*$ and updates the target identity public key with the new value $pk'_{ID^*} = (pk'_1, pk'_2) = (g_1^x, g_1^y)$. Note that $\mathcal{A}_1$ cannot request the target identity partial private key from $\mathcal{C}$.

Stage 3. Inputting a message m^*, pp, pk'_{ID^*}, ID^*, the attacker $\mathcal{A}_1$:

- calculates $d^* = H_d(ID^*)$, $e^* = H_e(m^*, ID^*, pk'_{ID^*})$.
- sets $\mathcal{D}^* = \{i | d^*[i] = 1\}$ and $\mathcal{E}^* = \{i | e^*[i] = 1\}$, where $d^*[i]$ and $e^*[i]$ stand for the ith bit of d^* and e^*.
- chooses randomly $r_d, r_e \in \mathbb{Z}_p^*$ and calculates
$$\begin{aligned} \sigma &= (\sigma_1, \sigma_2, \sigma_3) \\ &= (g_2^{x^{-1}} g_3^{yx^{-1}} D^{r_d} E^{r_e}, g_1^{xr_d}, g_1^{xr_e}), \end{aligned}$$
where $D = d' \prod_{i \in \mathcal{D}^*} d_i$, $E = e' \prod_{i \in \mathcal{E}^*} e_i$.
- outputs σ as the forged signature on m^* under pk'_{ID^*}.

Obviously, the forgery σ is sound on m^* under ID^* with pk'_{ID^*} since

$$\begin{aligned} \hat{e}(\sigma_1, pk'_1) &= \hat{e}(g_2^{x^{-1}} g_3^{yx^{-1}} D^{r_d} E^{r_e}, g_1^x) \\ &= \hat{e}(g_2^{x^{-1}}, g_1^x) \cdot \hat{e}(g_3^{yx^{-1}}, g_1^x) \cdot \hat{e}(D^{r_d}, g_1^x) \cdot \hat{e}(E^{r_e}, g_1^x) \\ &= \hat{e}(g_2, g_1) \cdot \hat{e}(g_3, g_1^y) \cdot \hat{e}(D, g_1^{xr_d}) \cdot \hat{e}(E, g_1^{xr_e}) \\ &= Z \cdot \hat{e}(g_3, pk'_2) \cdot \hat{e}(D, \sigma_2) \cdot \hat{e}(E, \sigma_3), \end{aligned}$$

where $Z = \hat{e}(g_1, g_2)$.

2) MALICIOUS-BUT-PASSIVE KGC ATTACKS

Here, we will illustrate that the KGC without knowing the user secret value v_{ID} can impersonate any user ID to give a valid forgery σ on any message m under ID with pk_{ID}. The details are as follows.

Stage 1. On the basis of $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, g)$, the KGC initializes the systems as follows:

- chooses randomly $\alpha, \beta, \gamma, x', x_1, x_2, \ldots, x_{n_d}, y', y_1, y_2, \ldots, y_{n_e}$ from $\mathbb{Z}_p^*$ and sets $g_1 = g^\alpha, g_2 = g^\beta, g_3 = g^\gamma, Z = \hat{e}(g_1, g_2), d' = g^{x'}, \vec{d} = \{d_i\}_{i=1}^{n_d} = \{g^{x_i}\}_{i=1}^{n_d}, e' = g^{y'}, \vec{e} = \{e_i\}_{i=1}^{n_e} = \{g^{y_i}\}_{i=1}^{n_e}$.
- selects two concrete cryptographic hashing $H_d : \{0,1\}^* \to \{0,1\}^{n_d}$, and $H_e : \{0,1\}^* \to \{0,1\}^{n_e}$, where n_d and n_e are fixed lengths.
- opens $pp = (\mathbb{G}_1, \mathbb{G}_2, g, g_1, g_2, g_3, d', \vec{d}, e', \vec{e}, Z, H_d, H_e)$ as the public parameters and keeps these trapdoors $(\alpha, \beta, \gamma, x', x_1, x_2, \ldots, x_{n_d}, y', y_1, y_2, \ldots, y_{n_e})$ secret.

Note that it is impossible for any PPT third party to detect the above trapdoors embedded in the public parameters provided by the KGC due to the **DL** assumption.

Stage 2. Taking as input pp, the entity with the identity ID performs the **UKG** algorithm to set the identity secret value $v_{ID} = (\tau, x)$ $(\in_R \mathbb{Z}_p)$ and the corresponding public key as $pk_{ID} = (pk_1, pk_2) = (g^\tau, g^x)$. Here, v_{ID} is kept a secret from the KGC.

Stage 3. Given pp, pk_{ID}, m and σ, the KGC:

- computes $d = H_d(ID)$, $e = H_e(m, ID, pk_{ID})$.
- sets $\mathcal{D} = \{i | d[i] = 1\}$ and $\mathcal{E} = \{i | e[i] = 1\}$, where $d[i]$ and $e[i]$ stand for the ith bit of d and e, respectively.
- lets $D = d' \prod_{i \in \mathcal{D}} d_i = g^a$, $E = e' \prod_{i \in \mathcal{E}} e_i = g^b$, where $a = x' \sum_{i \in \mathcal{D}} x_i$, $b = y' \sum_{i \in \mathcal{E}} y_i$
- chooses randomly $r_d, r_e \in \mathbb{Z}_p^*$ and calculates

$$\begin{aligned}\sigma &= (\sigma_1, \sigma_2, \sigma_3)\\ &= (g^{ar_d + br_e}, pk_2^{-\frac{\gamma}{a}} pk_1^{r_d}, g^{-\frac{\alpha\beta}{b}} pk_1^{r_e}).\end{aligned}$$

- outputs σ as her/his forged signature on m under pk_{ID}.

Obviously, the forgery σ given by KGC is sound on the message m under the target identity ID with $pk_{ID} = (pk_1, pk_2)$ since

$$\begin{aligned}&Z \cdot \hat{e}(pk_2, g_3) \cdot \hat{e}(D, \sigma_2) \cdot \hat{e}(E, \sigma_3)\\ &= \hat{e}(g_1, g_2) \cdot \hat{e}(pk_2, g^\gamma) \cdot \hat{e}(D, pk_2^{-\frac{\gamma}{a}} pk_1^{r_d}) \cdot \hat{e}(E, g^{-\frac{\alpha\beta}{b}} pk_1^{r_e})\\ &= \hat{e}(g^\alpha, g^\beta) \cdot \hat{e}(pk_2, g^\gamma) \cdot \hat{e}(g^a, pk_2^{-\frac{\gamma}{a}} pk_1^{r_d}) \cdot \hat{e}(g^b, g^{-\frac{\alpha\beta}{b}} pk_1^{r_e})\\ &= \hat{e}(g^a, pk_1^{r_d}) \cdot \hat{e}(g^b, pk_1^{r_e})\\ &= \hat{e}(g^{ar_d + br_e}, pk_1)\\ &= \hat{e}(\sigma_1, pk_1).\end{aligned}$$

IV. OUR CLS SCHEME

Here, we come up with a CLS scheme which can stop KGC forging a false key pair for a target user without being detected by the victim, and reduce its security to two classic signature schemes [31], [32], and cryptographic hashing in the standard model.

A. BUILDING BLOCKS

Now, we take a brief look at the classical signature schemes [31], [32] to make our concrete construction and its security proofs more clear.

Waters' Digital Signature (WDS) [31]

- **SetUKey.** Given an instance $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, g)$, a user:
  - picks randomly β from $\mathbb{Z}_p^*$ and calculates $h_1 = g^\beta$.
  - chooses two random elements h_2, w' from $\mathbb{G}_1$ and a random n_m-length vector $\vec{w} = \{w_i\}_{i=1}^{n_m}$ whose elements are also chosen from $\mathbb{G}_1$.
  - selects a cryptographic hashing $H_w : \{0,1\}^* \to \{0,1\}^{n_m}$.
  - needs to make $sk_{ID} = h_2^\beta$ private and $pk_{ID} = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, g, h_1, h_2, w', \vec{w})$ public.
- **Signing.** Inputting a message m and pk_{ID}, the user:
  - picks randomly r_m from $\mathbb{Z}_p^*$ and calculates $w = H_w(pk_{ID} \| m)$.
  - sets $\mathcal{W} = \{i | w[i] = 1, i = 1, 2, \ldots, n_m\}$, where $w[i]$ stands for the ith bit of w.
  - generates a signature as follows:

$$\sigma = (\sigma_1, \sigma_2) = (h_2^\beta (W)^{r_m}, g^{r_m}),$$

    where $W = w' \prod_{i \in \mathcal{W}} w_i$.
- **Verifying.** Inputting pk_{ID}, m and σ, a verifier:
  - calculates $w = H_w(pk_{ID} \| m)$ and sets $\mathcal{W} = \{i | w[i] = 1, i = 1, 2, \ldots, n_m\}$, where $w[i]$ stands for the ith bit of w.
  - parses σ as (σ_1, σ_2) and checks the verification equation below:

$$\hat{e}(\sigma_1, g) \stackrel{?}{=} \hat{e}(h_1, h_2)\hat{e}(W, \sigma_2),$$

    where $W = w' \prod_{i \in \mathcal{W}} w_i$.
  - outputs "TRUE" if the above formula holds; otherwise, outputs "FALSE".

Paterson *et al.*'s Identity-Based Signature (PIBS) [32]

- **Setup.** On the basis of $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, g)$, KGC:
  - picks randomly α from $\mathbb{Z}_p^*$ and calculates $g_1 = g^\alpha$.
  - chooses random elements g_2, u', v' from $\mathbb{G}_1$ and two vectors $\vec{u} = \{u_i\}_{i=1}^{n_u}$, $\vec{v} = \{v_i\}_{i=1}^{n_m}$ of length n_u and n_m, respectively. Note that, all values are randomly chosen from $\mathbb{G}_1$.
  - selects two cryptographic hashing $H_u : \{0,1\}^* \to \{0,1\}^{n_u}$, and $H_v : \mathbb{G}_1 \to \{0,1\}^{n_m}$, where n_u and n_m are fixed lengths.
  - needs to respectively make $msk = g_2^\alpha$ private and $pp = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, g, g_1, g_2, u', \vec{u}, v', \vec{v}, H_u, H_v)$ public.
- **Extracting.** Inputting an identity ID, KGC:
  - calculates $u = H_u(ID)$ and sets $\mathcal{U} = \{i | u[i] = 1\}$, where $u[i]$ stands for the ith bit of u.
  - picks $r_u \in_R \mathbb{Z}_p^*$ and computes

$$sk_{ID} = (sk_1, sk_2) = (g_2^\alpha (U)^{r_u}, g^{r_u}),$$

    where $U = u' \prod_{i \in \mathcal{U}} u_i$.
  - sends the secret key sk_{ID} to the user ID securely.
- **Signing.** Inputting pp, ID, a message m, the user:
  - parses sk_{ID} as (sk_1, sk_2) and calculates $v = H_v(ID \| pp \| m \| sk_2)$.

- sets $\mathcal{V} = \{i|v[i] = 1, i = 1, 2, \ldots, n_m\}$, where $v[i]$ stands for the ith bit of v.
- picks randomly r_m from $\mathbb{Z}_p^*$ and generates a signature as follows:

$$\begin{aligned}\sigma &= (\sigma_1, \sigma_2, \sigma_3)\\ &= (sk_1(V)^{r_m}, sk_2, g^{r_m})\\ &= (g_2^{\alpha}(U)^{r_u}(V)^{r_m}, g^{r_u}, g^{r_m}),\end{aligned}$$

where $U = u' \prod_{i\in\mathcal{U}} u_i$, $V = v' \prod_{i\in\mathcal{V}} v_i$.

- **Verifying.** Inputting pp, ID, m and σ, a verifier:
  - parses σ as $(\sigma_1, \sigma_2, \sigma_3)$ and computes $u = H_u(ID)$ and $v = H_v(ID\|pp\|m\|\sigma_2)$.
  - sets $\mathcal{U} = \{i|u[i] = 1, i = 1, 2, \ldots, n_u\}$ and $\mathcal{V} = \{i|v[i] = 1, i = 1, 2, \ldots, n_m\}$, where $u[i]$ and $v[i]$ stand for the ith bit of u and v, respectively.
  - checks the verification equation below:

$$\hat{e}(\sigma_1, g) \stackrel{?}{=} \hat{e}(g_1, g_2)\hat{e}(U, \sigma_2)\hat{e}(V, \sigma_3),$$

where $U = u' \prod_{i\in\mathcal{U}} u_i$, $V = v' \prod_{i\in\mathcal{V}} v_i$.
  - outputs "TRUE" if the above formula holds; otherwise, outputs "FALSE".

B. OUR CONCRETE SCHEME

Here, our CLS scheme is formalized as follows.

- **Setup.** On the basis of $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, g)$, KGC:
  - chooses randomly α_1, α_2, x', x_1, x_2, $\ldots$, x_{n_u}, y', $y_1, y_2, \ldots, y_{n_m}$ from $\mathbb{Z}_p^*$ and calculates $g_1 = g^{\alpha_1}$, $g_2 = g^{\alpha_2}$, $u' = g^{x'}$, $\vec{u} = \{u_i\}_{i=1}^{n_u} = \{g^{x_i}\}_{i=1}^{n_u}$, $v' = g^{y'}$, $\vec{v} = \{v_i\}_{i=1}^{n_m} = \{g^{y_i}\}_{i=1}^{n_m}$.
  - selects three cryptographic hashing $H_u : \{0, 1\}^* \to \{0, 1\}^{n_u}$, $H_v : \{0, 1\}^* \to \{0, 1\}^{n_m}$, and $H_w : \{0, 1\}^* \to \{0, 1\}^{n_m}$, where n_u and n_m are fixed lengths.
  - needs to make $pp = (\mathbb{G}_1, \mathbb{G}_2, g, g_1, g_2, u', \vec{u}, v', \vec{v}, H_u, H_v, H_w)$ public and $msk = (\alpha_1, \alpha_2, x', x_1, x_2, \ldots, x_{n_u}, y', y_1, y_2, \ldots, y_{n_m})$ private.
- **UserKeyGenerating (UKG).** Taking as input an identity ID, a user:
  - chooses randomly $\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m}$ from $\mathbb{Z}_p^*$ and sets the identity secret value $e_{ID} = (\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m})$.
  - calculates $h_1 = g^{\beta_1}$, $h_2 = g^{\beta_2}$, $w' = g^{z'}$, $\vec{w} = \{w_i\}_{i=1}^{n_m} = \{g^{z_i}\}_{i=1}^{n_m}$ and sets the corresponding public key as $pk_{ID} = (h_1, h_2, w', \vec{w})$.
- **PartialPrivateKeyExtracting (PPKE).** Inputting pk_{ID} and ID, KGC:
  - calculates $u = H_u(pk_{ID}\|ID)$ and sets $\mathcal{U} = \{i|u[i] = 1\}$, where $u[i]$ stands for the ith bit of u.
  - picks $r_u \in_R \mathbb{Z}_p^*$ and calculates

$$d_{ID} = (d_1, d_2) = (g_2^{\alpha_1}(U)^{r_u}), g^{r_u})$$

where $U = u' \prod_{i\in\mathcal{U}} u_i$.
  - transmits securely the partial private key d_{ID} to the user ID. In fact, $sk_{ID} = (d_{ID}, e_{ID})$ stands for the user full secret key.
- **Signing.** Inputting pp, ID, pk_{ID} and a message m, the user:
  - parses sk_{ID} as $(d_{ID}, e_{ID}) = (d_1, d_2, \beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m})$.
  - computes $v = H_v(pk_{ID}\|ID\|pp\|m\|d_2)$ and $w = H_w(pk_{ID}\|ID\|pp\|m\|d_2)$.
  - sets $\mathcal{V} = \{i|v[i] = 1, i = 1, 2, \ldots, n_m\}$ and $\mathcal{W} = \{i|w[i] = 1, i = 1, 2, \ldots, n_m\}$, where $v[i]$ and $w[i]$ stand for the ith bit of v and w, respectively.
  - selects randomly $r_m \in \mathbb{Z}_p^*$ and calculates

$$\begin{aligned}\sigma &= (\sigma_1, \sigma_2, \sigma_3)\\ &= (d_1 h_2^{\beta_1}(VW)^{r_m}, d_2, g^{r_m}),\end{aligned}$$

where $V = v' \prod_{i\in\mathcal{V}} v_i$ and $W = w' \prod_{i\in\mathcal{W}} w_i$.
- **Verifying.** Inputting pp, ID, pk_{ID}, m and σ, a verifier:
  - parses σ as $(\sigma_1, \sigma_2, \sigma_3)$ and computes $u = H_u(pk_{ID}\|ID)$, $v = H_v(pk_{ID}\|ID\|pp\|m\|\sigma_2)$ and $w = H_w(pk_{ID}\|ID\|pp\|m\|\sigma_2)$.
  - sets $\mathcal{U} = \{i|u[i] = 1, i = 1, 2, \ldots, n_u\}$, $\mathcal{V} = \{i|v[i] = 1, i = 1, 2, \ldots, n_m\}$ and $\mathcal{W} = \{i|w[i] = 1, i = 1, 2, \ldots, n_m\}$, where $u[i]$, $v[i]$ and $w[i]$ stand for the ith bit of u, v and w, respectively.
  - checks the verification equation:

$$\hat{e}(\sigma_1, g) \stackrel{?}{=} \hat{e}(g_1, g_2)\hat{e}(h_1, h_2)\hat{e}(U, \sigma_2)\hat{e}(VW, \sigma_3).$$

where $U = u' \prod_{i\in\mathcal{U}} u_i$, $V = v' \prod_{i\in\mathcal{V}} v_i$ and $W = w' \prod_{i\in\mathcal{W}} w_i$.
  - outputs "TRUE" if the above formula holds; otherwise, outputs "FALSE".

Correctness of the scheme We set U, V, W in the same methods as above and have that

$$\begin{aligned}\hat{e}(\sigma_1, g) &= \hat{e}(d_1 h_2^{\beta_1}(VW)^{r_m}, g)\\ &= \hat{e}(g_2^{\alpha_1} U^{r_u} h_2^{\beta_1}(VW)^{r_m}, g)\\ &= \hat{e}(g_2^{\alpha_1}, g)\hat{e}(U^{r_u}, g)\hat{e}(h_2^{\beta_1}, g)\hat{e}((VW)^{r_m}, g)\\ &= \hat{e}(g_2, g^{\alpha_1})\hat{e}(U, g^{r_u})\hat{e}(h_2, g^{\beta_1})\hat{e}(VW, g^{r_m})\\ &= \hat{e}(g_1, g_2)\hat{e}(h_1, h_2)\hat{e}(U, \sigma_2)\hat{e}(VW, \sigma_3).\end{aligned}$$

In reality, we can concisely set $msk = \alpha_1$ and $e_{ID} = \beta_1$ because the others are not used during the execution of the proposed CLS scheme. Here, all of them are explicitly listed to implement the following argument easily.

C. SECURITY ANALYSIS

We present the three lemmas to argue the above construction security without relying on random oracles.

Lemma 1: If **PIBS** is existential unforgeable, the proposed scheme is existential unforgeable against public key replacement attacker $\mathcal{A}_1$.

Proof: If a PPT attacker $\mathcal{A}_1$ can penetrate our CLS scheme, then a PPT attacker $\mathcal{B}_1$ who can break the **PIBS** scheme can be simulated with a non-negligible probability. In addition, the attacker $\mathcal{B}_1$ will maintain a list T to record those interaction information with the attacker $\mathcal{A}_1$ in the whole process.

Init. The attacker $\mathcal{B}_1$ adopts $pp = (\mathbb{G}_1, \mathbb{G}_2, g, g_1, g_2, u', \vec{u}, v', \vec{v}, H_u, H_v, H_w)$ from **PIBS** to initialize the system and returns it to the attacker $\mathcal{A}_1$ as the system parameters. Note that H_u, H_v, H_w denote secure hash functions, where H_w is separately picked by the attacker $\mathcal{B}_1$.

Queries. At this stage, the attacker $\mathcal{A}_1$ can adaptively do some queries and the attacker $\mathcal{B}_1$ responds them as follows:

- $\mathcal{O}^{ppk}(ID, pk_{ID})$: The attacker $\mathcal{B}_1$ invokes the algorithm **PIBS.Extracting** to derive the partial private key d_{ID} related to the item (ID, pk_{ID}) for the attacker $\mathcal{A}_1$.
- $\mathcal{O}^{sv}(ID)$: The attacker $\mathcal{B}_1$ searches the secret value $e_{ID} = (\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m})$ related to the item ID from the list T. If the search fails, the attacker $\mathcal{B}_1$ first picks $(\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m}) \in \mathbb{Z}_p^*$ and stores these values as the corresponding secret value in the list T. At last, the attacker $\mathcal{B}_1$ returns e_{ID} to the attacker $\mathcal{A}_1$.
- $\mathcal{O}^{pk}(ID)$: The attacker $\mathcal{B}_1$ retrieves the public key $pk_{ID} = (h_1, h_2, w', w_1, \ldots, w_{n_m})$ related to ID from the list T. If the retrieval fails, the attacker $\mathcal{B}_1$ first picks $(\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m})$ from $\mathbb{Z}_p^*$ like in $\mathcal{O}^{sv}(ID)$ and then sets $h_1 = g^{\beta_1}, h_2 = g^{\beta_2}, v' = g^{z'}, v_1 = g^{z_1}, \ldots, v_{n_m} = g^{z_{nm}}$. At last, the attacker $\mathcal{B}_1$ returns pk_{ID} to the attacker $\mathcal{A}_1$.
- $\mathcal{O}^{rep}(ID, pk'_{ID})$: The attacker $\mathcal{B}_1$ updates the public key pk_{ID} related to the identity ID with the new value pk'_{ID} provided by the attacker $\mathcal{A}_1$ in the list T. If these item related to the identity ID has not been established, the attacker $\mathcal{B}_1$ directly sets the user public key to be pk'_{ID}.
- $\mathcal{O}^{sign}(ID, pk_{ID}, m)$: The attacker $\mathcal{B}_1$ first retrieves the list T to obtain the secret value $e_{ID} = (\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m})$ related to the identity ID. Then, the attacker $\mathcal{B}_2$ requests the underlying algorithm **PIBS.Signing** to obtain a temporary tuple $(\sigma'_1, \sigma'_2, \sigma'_3)$ on the requested message m under the designated identity $pk_{ID}\|ID$. Next, the attacker $\mathcal{B}_1$ calculates $w = H_w(pk_{ID}\|ID\|pp\|m\|\sigma'_2)$ and sets $\sigma_1 = \sigma'_1 h_2^{\beta_1} (\sigma'_3)^{\sum_{i\in\mathcal{W}} z_i}$ where $\mathcal{W} = \{i|w[i] = 1, i = 1, 2, \ldots, n_m\}$. At last, the attacker $\mathcal{B}_2$ sets $\sigma = (\sigma_1, \sigma_2, \sigma_3) = (\sigma_1, \sigma'_2, \sigma'_3)$ and returns it to $\mathcal{A}_2$ as the signature on m under ID with pk_{ID}.

Forgery. If the attacker $\mathcal{A}_1$ takes m^*, ID^*, pk_{ID^*} as input and eventually returns a valid signature $\sigma^* = (\sigma_1^*, \sigma_2^*, \sigma_3^*)$, then the attacker $\mathcal{B}_1$ is surely able to break the **PIBS** scheme by giving a valid forgery $\tilde{\sigma}$ on m^* under the designated identity $pk_{ID^*}\|ID^*$ as follows:

- computes $w^* = H_w(pk_{ID^*}\|ID^*\|pp\|m^*\|\sigma_2^*)$.
- lets $\mathcal{W}^* = \{i|w^*[i] = 1, i = 1, 2, \ldots, n_m\}$, where $w^*[i]$ stands for the ith bit of w^*.
- sets $\tilde{\sigma}_1 = \sigma_1^*/((h_2^*)^{\beta_1^*}(\sigma_3^*)^{\sum_{i\in\mathcal{V}^*} z_i^*})$, $\tilde{\sigma}_2 = \sigma_3^*$.

Obviously, the above result is incompatible with that **PIBS** is existential unforgeable [32]. Therefore, the construction is existential unforgeable against the attacks from the public key replacement adversary.

TABLE 1. Security analysis.

Scheme	Type I	Type II	Level 3
[27]	√	×	√
[4]	×	×	×
Ours	√	√	√

Lemma 2: If the **WDS** scheme is existential unforgeable, our CLS scheme is existential unforgeable against malicious-but-passive KGC $\mathcal{A}_2$.

Proof: If a PPT attacker $\mathcal{A}_2$ can penetrate our CLS scheme, then a PPT attacker $\mathcal{B}_2$ who can break the **WDS** scheme can be simulated with a non-negligible probability. In addition, the attacker $\mathcal{B}_2$ will maintain a list T to record those interaction information with the attacker $\mathcal{A}_2$ in the whole process.

Init. The attacker $\mathcal{A}_2$ adaptively sets the system parameters (msk, pp) and transmits them to the challenger $\mathcal{C}$. Note that, $msk = (\alpha_1, \alpha_2, x', x_1, x_2, \ldots, x_{n_u}, y', y_1, y_2, \ldots, y_{n_m})$ and $pp = (\mathbb{G}_1, \mathbb{G}_2, g, g_1, g_2, u', \vec{u}, v', \vec{v}, H_u, H_v, H_w)$, where $g_1 = g^{\alpha_1}, g_2 = g^{\alpha_2}, u' = g^{x'}, \vec{u} = \{u_i\}_{i=1}^{n_u} = \{g^{x_i}\}_{i=1}^{n_u}$, $v' = g^{y'}, \vec{v} = \{v_i\}_{i=1}^{n_m} = \{g^{y_i}\}_{i=1}^{n_m}$.

Queries. At this stage, the attacker $\mathcal{A}_2$ can adaptively do some queries and the attacker $\mathcal{B}_2$ responds them as follows:

- $\mathcal{O}^{sv}(ID)$: The attacker $\mathcal{B}_2$ searches the secret value $e_{ID} = (\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m})$ related to the item ID from the list T. If the search fails, the attacker $\mathcal{B}_2$ first picks $(\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m}) \in \mathbb{Z}_p^*$ and stores these values as the corresponding secret value in the list T. At last, the attacker $\mathcal{B}_2$ returns e_{ID} to the attacker $\mathcal{A}_2$.
- $\mathcal{O}^{pk}(ID)$: The attacker $\mathcal{B}_2$ retrieves the public key $pk_{ID} = (h_1, h_2, w', w_1, \ldots, w_{n_m})$ related to ID from the list T. If the retrieval fails, the attacker $\mathcal{B}_2$ first picks $(\beta_1, \beta_2, z', z_1, z_2, \ldots, z_{n_m})$ from $\mathbb{Z}_p^*$ like in $\mathcal{O}^{sv}(ID)$ and then sets $h_1 = g^{\beta_1}, h_2 = g^{\beta_2}, v' = g^{z'}, v_1 = g^{z_1}, \ldots, v_{n_m} = g^{z_{nm}}$. At last, the attacker $\mathcal{B}_2$ returns pk_{ID} to the attacker $\mathcal{A}_2$.
- $\mathcal{O}^{rep}(ID, pk'_{ID})$: The attacker $\mathcal{B}_2$ updates the public key pk_{ID} related to the identity ID with the new value pk'_{ID} provided by the attacker $\mathcal{A}_2$ in the list T. If these item related to the identity ID has not been established, the attacker $\mathcal{B}_2$ directly sets the user public key to be pk'_{ID}.
- $\mathcal{O}^{sign}(ID, pk_{ID}, m)$: Inputting msk, the attacker $\mathcal{B}_2$ first simulates the algorithm **PPKE** to obtain the partial private key $d_{ID} = (d_1, d_2) = (g_2^{\alpha_1}(U)^{r_u}, g^{r_u})$ related to the identity ID. Then, the attacker $\mathcal{B}_2$ inquires the underlying algorithm **WDS.Signing** to obtain a temporary tuple (σ'_1, σ'_2) on the designated message $ID\|pp\|m\|d_2$ under the requested public key pk_{ID}. Next, the attacker $\mathcal{B}_2$ computes $v = H_v(pk_{ID}\|ID\|pp\|m\|d_2)$ and sets $\sigma_1 = \sigma'_1 d_1 (\sigma'_2)^{\sum_{i\in\mathcal{V}} y_i}$ where $\mathcal{V} = \{i|v[i] = 1, i = 1, 2, \ldots, n_m\}$. At last, the attacker $\mathcal{B}_2$ sets $\sigma = (\sigma_1, \sigma_2, \sigma_3) = (\sigma_1, d_2, \sigma'_2)$ and returns it to $\mathcal{A}_2$ as the signature on m under ID with pk_{ID}.

TABLE 2. Efficiency analysis.

Scheme	Signing	Verifying	$\lvert\sigma\rvert$
[27]	$7T_E + (n_m + 2)T_M$	$7T_P + 3T_{\bar{M}} + T_E + (2n_u + n_m + 1)T_M$	$5\lvert\mathbb{G}_1\rvert$
[4]	$5T_E + (n_u + n_m + 5)T_M$	$4T_P + 3T_{\bar{M}} + T_E + (n_u + n_m + 1)T_M$	$3\lvert\mathbb{G}_1\rvert$
Ours	$3T_E + (2n_m + 3)T_M$	$3T_P + 3T_{\bar{M}} + (n_u + 2n_m + 1)T_M$	$3\lvert\mathbb{G}_1\rvert$

T_E: Scalar exponent operating time in $\mathbb{G}_1$; T_M: Multiplication operating time in $\mathbb{G}_1$;
$T_{\bar{M}}$: Multiplication operating time in $\mathbb{G}_2$; T_P: Pairing operating time;
n_u, n_m: The expected length of hashing output.

Forgery. If the attacker $\mathcal{A}_2$ takes m^*, ID^*, pk_{ID^*} as input and eventually returns a valid signature $\sigma^* = (\sigma_1^*, \sigma_2^*, \sigma_3^*)$, then the attacker $\mathcal{B}_2$ is surely able to break the **WDS** scheme by providing a valid forgery $\tilde{\sigma}$ on the designated message $ID^*\|pp\|m^*\|\sigma_2^*$ under pk_{ID^*} as follows:

- computes $v^* = H_v(pk_{ID^*}\|ID^*\|pp\|m^*\|\sigma_2^*)$.
- lets $\mathcal{V}^* = \{i|v^*[i] = 1, i = 1, 2, \ldots, n_m\}$, where $v^*[i]$ is the ith bit of v^*.
- sets $\tilde{\sigma}_1 = \sigma_1^*/(d_1^*(\sigma_3^*)^{\sum_{i\in\mathcal{V}^*} y_i})$, $\tilde{\sigma}_2 = \sigma_3^*$.

Obviously, the above result is incompatible with that **WDS** is existential unforgeable [31]. Therefore, the proposed scheme is existential unforgeable against the attacks from malicious-but-passive KGC.

Lemma 3: If H_u is cryptographic hash function, our construction is able to withstand the Level 3 attacker $\mathcal{A}_3$.

Analysis: $\mathcal{A}_3$ breaking our CLS scheme means that $\mathcal{A}_3$ can give a valid key pair (pk'_{ID^*}, sk'_{ID^*}) and the target user ID^* has no evidence to deny this key pair. In other words, the target user holds the same partial private key corresponding to pk'_{ID^*} and pk_{ID^*}. It implies $H_u(pk'_{ID^*}\|ID^*) = H_u(pk_{ID^*}\|ID^*)$. Obviously, it is incompatible with that H_u is cryptographic hashing. Therefore, the proposed scheme can repudiate the Level 3 attacks.

It is obvious that our construction security can be guaranteed by the above three lemmas without relying on random oracles.

V. COMPARISON

In [27], Tseng *et al.* made a comprehensive summary about the previous classical works [9]–[12], [14], [15], [22], [23], [25], [26], [28], [29] and gave a CLS scheme with the current whole optimum performance. Almost simultaneously, Shim also introduced an efficient CLS scheme. Here, we make a detailed comparison between our CLS scheme with the two typical ones [4], [27] in security properties and efficiency.

By contrast, we find that [27] meets the property of Girault's level-3 security like ours but has longer signature length, and [4] has the same signature size with our CLS scheme but cannot withstand any attack launched by Type I, Type II and Level 3. In summary, our scheme not only overcomes the weaknesses in [4], [27], but also has efficient signing and verifying, shorter length of signature. More detailed comparisons between [4], [27] and our scheme are illustrated in Table 1 and 2.

Note that, the running time of the different operations from the PCB library are stable on a given platform. For example, an optimal-ate pairing operation takes 0.524 ms on Phenom II X4 940, 3.0 GHZ, which has been validated in [4]. Therefore, the numbers of each operation listed in Table 2 can reflect the execution cost of each scheme.

VI. CONCLUSION

In this paper, our analysis indicated that Shim's construction is not immune to the public key replacement adversaries and the malicious-but-passive KGC. To repair these weaknesses, we constructed a top-level CLS scheme and proved its security against the Type 1, Type 2 and Level 3 attacks without relying on ROM. The proposed scheme has shorter signature length, and lower computation and verification cost compared with Shim's scheme.

REFERENCES

[1] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. Workshop Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1984, pp. 47–53.

[2] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proc. 9th Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Taipei, Taiwan, Nov. 2003, pp. 452–473.

[3] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," *J. ACM*, vol. 51, no. 4, pp. 557–594, Jul. 2004.

[4] K.-A. Shim, "A new certificateless signature scheme provably secure in the standard model," *IEEE Syst. J.*, vol. 13, no. 2, pp. 1421–1430, Jun. 2019.

[5] X. Huang, Y. Mu, W. Susilo, D. S. Wong, and W. Wu, "Certificateless signature revisited," in *Proc. 12th Australas. Conf. Inf. Secur. Privacy (ACISP)*, Townsville, QLD, Australia, Jul. 2007, pp. 308–322.

[6] Z. Zhang, D. S. Wong, J. Xu, and D. Feng, "Certificateless public-key signature: Security model and efficient construction," in *Proc. 4th Int. Conf. Appl. Cryptogr. Netw. Secur. (ACNS)*, Singapore, Jun. 2006, pp. 293–308.

[7] Z. Guan, Y. Zhang, L. Zhu, L. Wu, and S. Yu, "EFFECT: An efficient flexible privacy-preserving data aggregation scheme with authentication in smart grid," *Sci. China Inf. Sci.*, vol. 62, no. 3, Mar. 2019, Art. no. 32103.

[8] R. Tso, X. Yi, and X. Huang, "Efficient and short certificateless signature," in *Proc. Int. Conf. Cryptol. Netw., Secur. (CANS)*, Hong Kong, Dec. 2008, pp. 64–79.

[9] H. Xiong, Z. Qin, and F. Li, "An improved certificateless signature scheme secure in the standard model," *Fundamenta Informaticae*, vol. 88, nos. 1–2, pp. 193–206, Dec. 2008.

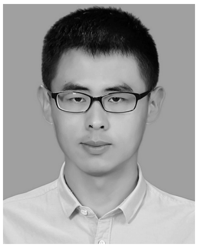
[10] J. K. Liu, J. Baek, W. Susilo, and J. Zhou, "Certificate-based signature schemes without pairings or random oracles," in *Proc. Int. Conf. Inf. Secur. (ISC)*, Taipei, Taiwan, Sep. 2008, pp. 285–297.

[11] H. Du and Q. Wen, "Efficient and provably-secure certificateless short signature scheme from bilinear pairings," *Comput. Standard Interfaces*, vol. 31, no. 2, pp. 390–394, Feb. 2009.

[12] C.-I. Fan, R.-H. Hsu, and P.-H. Ho, "Truly non-repudiation certificateless short signature scheme from bilinear pairings," *J. Inf. Sci. Eng.*, vol. 27, pp. 969–982, May 2011.

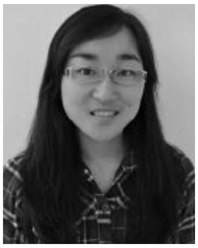
[13] D. He, J. Chen, and R. Zhang, "An efficient and provably-secure certificateless signature scheme without bilinear pairings," *Int. J. Commun. Syst.*, vol. 25, no. 11, pp. 1432–1442, Nov. 2012.

[14] C.-I. Fan, P.-H. Ho, and Y.-F. Tseng, "Strongly secure certificateless signature scheme supporting batch verification," *Math. Problems Eng.*, vol. 2014, Apr. 2014, Art. no. 854135.
[15] Y.-C. Chen, R. Tso, G. Horng, C.-I. Fan, and R.-H. Hsu, "Strongly secure certificateless signature: Cryptanalysis and improvement of two schemes," *J. Inf. Sci. Eng.*, vol. 31, no. 1, pp. 297–314, 2015.
[16] K.-H. Yeh, C. Su, K.-K. R. Choo, and W. Chiu, "A novel certificateless signature scheme for smart objects in the Internet-of-Things," *Sensors*, vol. 17, no. 5, p. 1001, May 2017.
[17] X. Jia, D. He, Q. Liu, and K.-K. R. Choo, "An efficient provably-secure certificateless signature scheme for Internet-of-Things deployment," *Ad Hoc Netw.*, vol. 71, pp. 78–87, Mar. 2018.
[18] A. Karati, S. H. Islam, and G. P. Biswas, "A pairing-free and provably secure certificateless signature scheme," *Inf. Sci.*, vol. 450, pp. 378–391, Jun. 2018.
[19] J. K. Liu, M. H. Au, and W. Susilo, "Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model," in *Proc. 2nd ACM Symp. Inf., Comput. Commun. Secur. (ASIACCS)*, Singapore, Mar. 2007, pp. 278–283.
[20] Y. Yuan, D. Li, L. Tian, and H. Zhu, "Certificateless signature scheme without random oracles," in *Proc. 3rd Int. Conf. Workshops Adv. Inf. Secur. Assurance (ISA)*, Jun. 2009, pp. 31–40.
[21] Q. Xia, C. Xu, and Y. Yu, "Key replacement attack on two certificateless signature schemes without random oracles," *Key Eng. Mater.*, vols. 439–440, no. 5, pp. 1606–1611, 2010.
[22] Y. Yu, Y. Mu, G. Wang, Q. Xia, and B. Yang, "Improved certificateless signature scheme provably secure in the standard model," *IET Inf. Secur.*, vol. 6, no. 2, pp. 102–110, Jun. 2012.
[23] Y. Yuan and C. Wang, "Certificateless signature scheme with security enhanced in the standard model," *Inf. Process. Lett.*, vol. 114, no. 9, pp. 492–499, Sep. 2014.
[24] L. Pang, Y. Hu, Y. Liu, K. Xu, and H. Li, "Efficient and secure certificateless signature scheme in the standard model," *Int. J. Commun. Syst.*, vol. 30, no. 5, Mar. 2017, Art. no. e3041.
[25] F. Wang and L. Xu, "Strongly secure certificateless signature scheme in the standard model with resisting malicious-but-passive KGC attack ability," *J. Inf. Sci. Eng.*, vol. 33, no. 4, pp. 873–889, Jul. 2017.
[26] W. Yang, J. Weng, W. Luo, and A. Yang, "Strongly unforgeable certificateless signature resisting attacks from malicious-but-passive KGC," *Secur. Commun. Netw.*, vol. 2017, Nov. 2017, Art. no. 5704865.
[27] Y.-F. Tseng, C.-I. Fan, and C.-W. Chen, "Top-level secure certificateless signature scheme in the standard model," *IEEE Syst. J.*, to be published.
[28] T.-T. Tsai, S.-S. Huang, and Y.-M. Tseng, "Secure certificateless signature with revocation in the standard model," *Math. Problems Eng.*, vol. 2014, Nov. 2014, Art. no. 728591.
[29] C. Zhou, "Certificateless signcryption scheme without random oracles," *Chin. J. Electron.*, vol. 27, no. 5, pp. 1002–1008, Sep. 2018.
[30] M. Girault, "Self-certified public keys," in *Proc. Workshop Theory Appl. Cryptograph. Techn. (EUROCRYPT)*, Brighton, U.K., Apr. 1991, pp. 490–497.
[31] B. Waters, "Efficient identity-based encryption without random oracles," in *Proc. 24th Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, Aarhus, Denmark, May 2005, pp. 114–127.
[32] K. G. Paterson and J. C. N. Schuldt, "Efficient identity-based signatures secure in the standard model," in *Proc. 11th Australas. Conf. Inf. Secur. Privacy (ACISP)*, Melbourne, QLD, Australia, Jul. 2006, pp. 207–222.



WENJIE YANG received the Ph.D. degree from the College of Cyber Security/College of Information Science and Technology, Jinan University, China, in 2018. He is currently a Lecturer with the Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Mathematics and Informatics, Fujian Normal University, China. His research interests include cryptography and information security.

SHANGPENG WANG received the M.S. degree from the School of Economics, Fujian Normal University, China, in 2011, where he is currently pursuing the Ph.D. degree with the College of Mathematics and Informatics. His research interests include cryptography and information security.



WEI WU received the Ph.D. degree from the School of Computer Science and Software Engineering, University of Wollongong, Australia, in 2011. She is currently an Associate Professor with the College of Mathematics and Informatics, Fujian Normal University, China. She has published over 20 research papers in refereed international conferences and journals. Her research interests include cryptography and information security.

YI MU received the Ph.D. degree from The Australian National University, in 1994. He was a Full Professor with the University of Wollongong, Australia. He joined Fujian Normal University, China, where he is currently a Full Professor with the Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Mathematics and Informatics. His current research interests include cryptography, network security, and computer security. He was the Editor-in-Chief of the *International Journal of Applied Cryptography* and serves as an Associate Editor or a Guest Editor for many international journals.