Design Principles of Secure Certificateless Signature and Aggregate Signature Schemes for IoT Environments

KYUNG-AH SHIM, (Member, IEEE)
National Institute for Mathematical Sciences, Yuseong-gu, Daejeon 34047, South Korea
e-mail: kashim@nims.re.kr

This work was supported by the National Institute for Mathematical Sciences funded by the Ministry of Science and Information and Communication Technologies (ICT) of South Korea under Grant B22710000.

ABSTRACT Certificateless cryptography resolves the certificate management problem of public-key cryptography and the key-escrow problem of identity-based cryptography. An aggregate signature scheme which allows to aggregate $k$ distinct signatures on $k$ distinct messages of $k$ distinct signers into a single signature reduces communication overhead and computational cost. Due to the suitability of certificateless signature (CLS) and certificateless aggregate signature (CLAS) schemes for IoT environments, similar CLS and CLAS schemes have been proposed for a long time and, despite their security proofs, they have been attacked and modified to prevent the attacks. Even now, similar design methods and similar attacks on the schemes are being repeated. In order to prevent the similar attacks on the schemes, it is necessary to analyze their causes and vulnerabilities. In this paper, we first show that recently proposed five CLS and CLAS schemes are insecure against universal forgery attacks, type I attacks, type II attacks or malicious-but-passive-KGC attacks. We discuss their security flaws, causes and countermeasures. We then present design principles to prevent various algebraic attacks including our attacks. The design principles will help in the construction of secure CLS and CLAS schemes against the previous attacks and potential attacks.

INDEX TERMS Aggregate signature, certificateless signature, malicious-but-passive-KGC attack, type I attack, type II attack, universal forgery attack.

I. INTRODUCTION
To ensure integrity, authenticity and non-repudiation, several kinds of public-key cryptography have been proposed. Al-Riyami and Paterson [1] proposed certificateless public-key cryptography to resolve the certificate management problem of public-key cryptography [2] and the key-escrow problem of identity (ID)-based cryptography [3]. In certificateless public-key cryptography, a user private key is computed by combining a partial private key generated by Key Generate Center (KGC) with a user-chosen secret. Certificateless public-key cryptography does not require public-key certificates and ciphertext/signatures are transmitted with users’ public keys. With these advantages, certificateless signature (CLS) schemes seem to be suitable for achieving authenticity and integrity of transmitted data in IoT environments [4], [5].

Since Al-Riyami and Paterson constructed the first certificateless scheme [1], there have been proposed many CLS schemes. Most of them turned out to be insecure against type I or type II attacks. In [6], Au et al. presented malicious-but-passive KGC attacks on several CLS schemes, in which the KGC could control the system parameters dishonestly to forge users’ valid signatures. Also, almost of the proposed CLS schemes are secure in the random oracle model. It is known that the schemes may not be secure [7], [8] when random oracles are instantiated with concrete hash functions in real applications. There are some attempts for constructing CLS scheme probably secure in the standard model by using the idea of Paterson-Schuldt’s ID-based signature scheme [9].
However, the schemes turned out to be insecure against the malicious-but-passive-KGC attacks, the type I attacks or type II attacks.

An aggregate signature scheme allows to aggregate $k$ distinct signatures on $k$ distinct messages into a single signature which reduces communication overhead and computational cost. In particular, the aggregation of multiple signatures is an indispensable technique to save channel bandwidth in IoT environments. Also, the verification of an aggregate signature of $k$ signatures is more efficient than the verifications of $k$ individual signatures. Since smart devices used in IoT environments are resource, energy and storage-constrained, and wireless, it is a challenging task to propose a compact aggregate signature scheme whose aggregate signature length is independent of the number of signers. Many provably secure certificateless aggregate signature (CLAS) schemes have been constructed [10], [11], [12], [13], [14], [15], [16], but most of them were broken or cryptanalyzed and then modified [17], [18], [19]. Recently, Deng et al. [20] and Tseng et al.’s scheme [21] proposed new efficient CLS and CLAS schemes provable secure in the standard model under the hardness assumptions of the mathematical problems.

Gentry [22] proposed a new concept of certificate-based cryptography (CBC) by combining the advantages of both public-key cryptography [2] and identity-based cryptography [3]. In the CBC scheme, a user computes a secret/public key pair and a Trust Authority (TA) issues the user’s certificate of the identity-public key pair by signing on the pair. To decrypt/sign a ciphertext/message, the user utilizes the secret key. This implicit use of certificate solve the certificate management problem in the public-key cryptography and the key-escrow problem in the identity-based cryptography. Recently, Verma et al. [23] proposed the first pairing-free compact certificate-based aggregate signature (CB-AS) scheme, CB-CAS, using a fixed state information. They showed that the scheme was secure against two types of adversaries under the intractability assumption of the mathematical problem.

Since Al-Riyami and Paterson’s scheme, a number of CLS and CLAS schemes have been proposed, but most of them were broken or cryptanalyzed. Some of them look similar since the schemes have been reproduced with minor modifications of the existing ones. In this paper, we show that recently proposed four CLS and CLAS schemes [16], [20], [21], and one CB-signature [23] scheme are insecure against universal forgery attacks, type I attacks, type II attacks or malicious-but-passive-KGC attacks. We then present design principles for construction secure CLS and CLAS schemes to prevent our attacks in Section IV. Section V concludes this paper.

II. DESCRIPTION OF RECENTLY PROPOSED CERTIFICATELESS SIGNATURE SCHEMES

Here, we describe four CLS and CLAS schemes [16], [20], [21], and one CB-signature scheme [23].

A. CLS AND CLAS SCHEMES IN STANDARD MODEL

We review four CLS and CLAS schemes, the KGC performs the followings:

1. Choose two groups, $\mathbb{G}_1$ and $\mathbb{G}_2$, with a prime order $q > 2v$, and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.
2. Select three secure hash functions $H_1, H_2, H_3 : \{0, 1\}^* \rightarrow \mathbb{Z}_q^*$.
3. Choose three generators $P, Q, Y$ of $\mathbb{G}_1$ and $x \in \mathbb{Z}_q^*$ calculate $P_{\text{pub}} = xP$, and set $\text{msk} = x$ as a master secret.
4. Publish the public system parameters $\text{params} = (\mathbb{G}_1, \mathbb{G}_2, q, e, P, Q, Y, P_{\text{pub}}, H_1, H_2, H_3)$.

- **PPK-Extract.** For the generation of a partial-private key of $ID_1$, the KGC performs the followings:
  1. Pick $r_i \in \mathbb{Z}_q^*$ and calculate a partial private key, $D_i = (R_i, d_i)$, where $R_i = r_iP$, $k_i = H_1(ID_1, R_i)$, $d_i = r_i + k_i x \mod q$.
  2. Send $D_i$ to the user in a secure way.

- **SV-Set.** A user with $ID_1$ chooses a secret value $t_i \in \mathbb{Z}_q^*$.

- **UPK-Set.** A user with $ID_1$ calculates $T_i = t_iP$ and sets $PK_i = (R_i, T_i)$ as a public key.

- **Sign.** A user with $ID_1$ generates a signature of a message $m_i$ as:
  1. Pick $u_i \in \mathbb{Z}_q^*$ and calculate $U_i = u_iP$, $h_i = H_2(m_i, U_i, ID_1, PK_i)$, $l_i = H_3(m_i, U_i, ID_1, PK_i)$, $V_i = (d_i + l_i)Q + h_i u_i Y$.
  2. Output $\sigma_i = (U_i, V_i)$ as a signature of $m_i$.

- **Verify.** For a signature $\sigma_i = (U_i, V_i)$ on $m_i$ under $\{ID_1, PK_i\}$, a verifier performs the followings:
  1. Compute $k_i = H_1(ID_1, R_i)$, $h_i = H_2(m_i, U_i, ID_1, PK_i)$, $l_i = H_3(m_i, U_i, ID_1, PK_i)$.
  2. Check the equality $e(V_i, P) = e(R_i + k_i P_{\text{pub}} + l_i T_i, Q)e(h_i U_i, Y)$. If the equality holds then accept it.
• **Agg.** For a tuple \((\sigma_1, \ldots, \sigma_n)\) on \(\{m_i\}_{i=1}^n\) for \(|\{ID_i, PK_i\}_{i=1}^n|\), an aggregator computes \(V = \sum_{i=1}^n V_i\) and outputs \(\sigma = (V, U_1, \ldots, U_n)\) as an aggregate signature.

• **Agg-Verify.** For an aggregate signature \(\sigma\) of \(\{m_i\}_{i=1}^n\) under \(|\{ID_i, PK_i\}_{i=1}^n|\), a verifier performs the followings:
  - Calculate
    \[k_i = H_1(ID_i, R_i), \quad h_i = H_2(m_i, U_i, ID_i, PK_i), \quad l_i = H_3(m_i, U_i, ID_i, PK_i), \quad i = 1, \ldots, n.\]
  - Check the equality
    \[e(V, P) = e(\sum_{i=1}^n (R_i + k_i P_{pub} + l_i T_i), Q) e(\sum_{i=1}^n h_i U_i, Y).\]
    If the equality holds, accept \(\sigma\).

In 1991, Girault [24] introduced Trusted Third Party’s three security levels: the stronger system security, the higher level of the KGC. Tseng et al. [21] designed a new CLS scheme based on bilinear parings achieving the level-3 security.

**Tseng et al.’s CLS Scheme**

**Setup.** To generate system parameters, the KGC does the followings:
  - Choose two secure hash functions, \(H_m : \{0, 1\}^* \rightarrow \{0, 1\}^{n_m}\) and \(H_u : \{0, 1\}^* \rightarrow \{0, 1\}^{n_u}\), for some \(n_m, n_u \in \mathbb{Z}\). The hash functions are utilized to get identities and messages of the fixed lengths.
  - Choose a bilinear map \(e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2\), pick \(\alpha \in \mathbb{Z}_p^*\) two elements \(g, g_2 \in \mathbb{G}_2\), and calculate \(g_1 = g^\alpha\).
  - Select a cryptographic hash function \(\tilde{H} : \mathbb{G}_1 \rightarrow \mathbb{G}_1\), \(\tilde{\nu}, \tilde{\nu} \in \mathbb{G}_1\) and vectors \((\tilde{m})_{i=1}^{n_m}\), \((\tilde{v}_i)_{i=1}^{n_v}\), \((\tilde{u}_i)_{i=1}^{n_u}\), \((\tilde{\nu}_i)_{i=1}^{n_v}\), \((\tilde{\tilde{u}}_i)_{i=1}^{n_u}\), \((\tilde{\tilde{\nu}}_i)_{i=1}^{n_v}\), \((\tilde{\tilde{\tilde{u}}}_i)_{i=1}^{n_u}\), where \(u_i, v_i, m_i \in \mathbb{G}_1\). Calculate
    \[m_i' = \tilde{H}(\tilde{m}), \quad v_i' = \tilde{H}(\tilde{\nu}), \quad u_i' = \tilde{H}(\tilde{u}), \quad \tilde{U} = (\tilde{u}_i = \tilde{H}(\tilde{u}_i))_{i=1}^{n_u}, \quad \tilde{V} = (\tilde{v}_i = \tilde{H}(\tilde{v}_i))_{i=1}^{n_v}, \quad \tilde{\tilde{M}} = (\tilde{m}_i = \tilde{H}(\tilde{m}_i))_{i=1}^{n_m}, \quad \tilde{\tilde{\tilde{M}}} = (\tilde{m}_i = \tilde{H}(\tilde{m}_i))_{i=1}^{n_m}.
    \]
  - Publish \(\text{params} = (g, \mathbb{G}_1, \mathbb{G}_2, e, g_1, g_2, u, v, m, \tilde{H}, H_u, H_v, H_a)\) as the system parameters and \(g_1^\alpha\) is a master secret.

• **Set-Secret Key.** A user chooses \(x \in \mathbb{Z}_p^*\) and sets \(r_{ID} = x\) as a secret key.

• **Set-Public Key.** A user with a secret key \(x\) computes \((pk_1, pk_2) = (g^x, g_1^x)\) as a public key.

• **Partial-Private Key.** The KGC computes a partial-private key of ID as:
  - Let \(u[i]\) and \(v[i]\) be the \(i\)-th bit of \(u = H_u(ID)\) and \(v = H_v(PK)\), respectively, where \(PK = (pk_1, pk_2)\).
  - Let \(U = \{i | u[i] = 1\} \subset \{1, 2, \ldots, n_u\}\) and \(V = \{i | v[i] = 1\} \subset \{1, 2, \ldots, n_v\}\) be the sets of indices \(i\) satisfying \(u[i] = 1\) and \(v[i] = 1\), respectively.
  - Choose \(r_x, r_y \in \mathbb{Z}_p^*\) and calculate a partial private key \(d_{ID}\) as
    \[d_{ID} = (d_1, d_2, d_3, d_4) = (g_1^{2x}(U_{ID})^y, g_2^{2x}(U_{PK})^{y'}, g^{a_x}, g^{a_y}),\]
    where \(U_{PK} = v' \prod_{i \in V} v_i \) and \(U_{ID} = u' \prod_{i \in U} u_i\).
  - At last, the KGC transmits \(d_{ID}\) to the user through a secure way.

• **Set-Private Key.** A full private key of ID is \(s_{ID} = (r_{ID}, d_{ID})\).

• **Sign.** A user with ID generates a signature of a message \(m_i\) as:
  - Pick \(r_m \in \mathbb{Z}_p^*\) and calculate \(m = H_m(m)\).
  - Let \(\mathcal{M} = \{[m[i] = 1] \subset \{1, \ldots, n_m\}\) be the set of indices \(i\) satisfying \(m[i] = 1\), where \(m[i] = \) \(i\)th bit of \(m = H_m(m)\).
  - Using its private key \(s_{ID} = (d_{ID}, r_{ID})\), compute a signature \(\sigma\) as
    \[\sigma = (\sigma_1, \sigma_2, R_u, R_v, R_m).
    \]
    where \(M = (\prod_{i \in \mathcal{M}} \tilde{m}_i)^x\).
  - **Verify.** Given a signature \(\sigma = (\sigma_1, \sigma_2, R_u, R_v, R_m)\) of \(m\) under \(|\{ID, PK\} = (pk_1, pk_2)|\), calculate \(U_{ID}, U_{PK}, M\) and check the equalities
    \[e(pk_1, g_1) = e(pk_2, g), \quad e(V_{12}, g) = e(g_2, pk_2^x) \cdot e(U_{ID}, R_v) \cdot e(U_{PK}, R_u) \cdot e(M, R_m).\]
    If the equalities hold then output \text{Valid.} Otherwise, output \text{Invalid.}

**B. CLASSES SCHEMES IN RANDOM ORACLE MODEL**

Mei et al. [16] constructed a CLS scheme for conditional privacy preservation in IoV.

**Mei et al.’s CLS Scheme**

• **Setup.** The TRA and the KGC perform the followings:
  - The KGC selects two groups, \(\mathbb{G} \) and \(\mathbb{G}_T\), with order \(p\), and a bilinear map \(e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T\).
  - The KGC selects four secure hash functions \(H_0, H_1, H_2 : \{0, 1\}^* \rightarrow \mathbb{G}, H_3 : \{0, 1\}^* \rightarrow \mathbb{Z}_p\).
  - The KGC chooses a generator \(P \in \mathbb{G}_1\) and \(s \in \mathbb{Z}_p\), calculates \(MPK = sP\), and keeps \(s\) as a master secret.
  - The TRA picks \(k \in \mathbb{Z}_p\), calculates \(T_{pub} = kP\), and keeps \(k\) as a tracking secret key.
  - Publish the system parameters \(pp = MPK, T_{pub}, e, \mathbb{G}, \mathbb{G}_T, H_0, H_1, H_2, H_3 > .\)

• **Pseudonym-Gen:** After a vehicle \(V_i\) joins VANET, the TRA generates a set of \(m\) pseudonyms as:
  - First, \(V_i\) chooses \(t_{ij} \in \mathbb{Z}_p\) and calculates
    \[PID_{i, j} = \{PID_{i, 1, j} = t_{ij} \} \in \mathbb{Z}_p^*\]
Then, $V_i$ sends $(ID_i, PID_i, t_i)$ to the TRA in a secure manner, where $ID_i$ is $V_i$’s real identity.

- After checking $ID_i$, the TRA calculates

$$PID_{i,2} = [ID_i \oplus H_0(TP_i, k \cdot PID_{i,1,1})]_{j \in [m]}$$

where $TP_i$ is a valid period of the pseudonym and outputs a set of the vehicle pseudonyms $PSUID_i = \{PSUID_i, (PID_{i,1,1}, PID_{i,2,2}, TP_{i,1})\}_{j \in [m]}$.

**Partial-Key-Gen:** The KGC calculates

$$Q_i = \{Q_{i,j} = H_1(PSUID_i)\}_{j \in [m]}$$

and outputs a set of the vehicle partial private keys $psk_i = \{psk_{i,j} = sQ_{i,j}\}_{j \in [m]}$.

**Vehicle-Key-Gen:** A vehicle $V_i$ chooses a secret key $vesk_i = x_i \in R \mathbb{Z}_p^*$ and calculates a public key $vepk_i = x_iP$.

**Sign:** After meeting a new RSU, a vehicle $V_i$ performs the following:

- For RSU’s identity $ID_{Rs}$, calculate $I_s = H_2(0, ID_{Rs})$, $I_s = H_2(1, ID_{Rs})$, sample a pseudo identity $PSUID_{i,j}$ from $PSUID_i$ and find the associated secret key $psk_i$ from $psk_i$.
- Calculate $E_i = psk_i + vesk_i$ and stores $E_i$ in a tamperproof device.
- To sign a message $msg_i$, $V_i$ selects $t_i$ as the timestamp, chooses $u_i \in \mathbb{Z}_p^*$ and calculates $U_i = u_iP$,

$$h_i = H_3(msg_i || t_i, vesk_i, PSUID_{i,j}, ID_{Rs}),$$

$$T_i = u_iJ_s + h_iE_i.$$

Then $V_i$ outputs a signature on $msg_i || |t_i, sig_i = (U_i, T_i)$ and sends

$$(msg_i, t_i, vesk_i, PSUID_{i,j}, ID_{Rs}, sig_i)$$

to the RSU. At last, $V_i$ cancels $PSUID_{i,j}$ from $PSUID_i$.

**Verify:** On receiving a signature $sig_i = (U_i, T_i)$ of $msg_i$ from $V_i$, an RSU confirms the timestamp. If it is valid then the RSU calculates

$$h_i = H_3(msg_i || |t_i, vesk_i, PSUID_{i,j}, ID_{Rs}),$$

and checks the equality

$$e(P, T_i) = e(MPK, h_iQ_i) e(h_ivesk_i, I_s) e(U_i, J_s).$$

If the equality holds, accept $sig_i$.

**Aggregate:** Given a set of signatures $sig_i = \{(U_i, T_i)\}_{i=1}^n$ of $\{msg_i, t_i, vesk_i, PSUID_{i,j}, ID_{Rs}\}_{i=1}^n$ from a set of vehicles $\{V_i\}_{i=1}^n$, a RSU calculates

$$U = \sum_{s=1}^n U_s, \quad T = \sum_{s=1}^n T_s,$$

and returns an aggregate signature $sig_i = (U, T)$. At last, the RSU sends

$$\{(msg_i, t_i, vesk_i, PSUID_{i,j}, ID_{Rs})\}_{i=1}^n$$
to a TMC.

**Aggregate-Verify:** For an aggregate signature $sig_i$ of the messages $\{msg_i\}_{i=1}^n$ for $n$ vehicles pseudonyms $\{PSUID_{i,j}\}_{i=1}^n$ and $n$ vehicle public keys $\{vepk_i\}_{i=1}^n$, the TMC does the following:

- Check the validity of $t_i$ for $s = 1, \ldots, n$. If they are valid, compute

$$h_i = H_3(msg_i || |t_i, vesk_i, PSUID_{i,j}, ID_{Rs}),$$

- Check the equality

$$e(P, T) = e(\sum_{i=1}^n h_ivepk_i, I_s)e(MPK, \sum_{i=1}^n h_iQ_i)e(U, J_s).$$

If the equality holds, accept $sig_i$.

### C. CB-AGGREGATE SIGNATURE SCHEME

Now, we describe Verma et al.’s certificate-based aggregate signature scheme, CB-CAS [23].

**Verma et al.’s Scheme.**

**Setup:** The TA does the following:

- Select a prime number $p$, a cyclic group $\mathbb{G}_T$ with order $p$ and a generator $P \in \mathbb{G}_T$.
- Choose $s \in R \mathbb{Z}_p^*$, compute $sP$ and set $pTA = sP$ and $sTA = s$ as the TA’s public key and secret key, respectively.
- Choose secure hash functions

$$H_0: \{0, 1\}^* \times \mathbb{G}_T \rightarrow \mathbb{Z}_p^*,$$

$$H_1: \{0, 1\}^* \times \mathbb{G}_T \times \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$$

and $\Delta \in \{0, 1\}^*$ as a random state information.
- Finally, the TA publishes

$$\Omega = <p, H_0, H_1, G_T, P, pTA, \Delta>$$

as the security parameters.

**UserKeyExtract:** A user $U_i$ with an identity $ID_i$ extracts a secret/public key pair as

- Choose $sID_i \in R \mathbb{Z}_p^*$ as a secret key and calculate $pID_i = sID_iP$ as a public key.

**CertExtract:** Through this interactive phase, a user receives a certificate from the TA.

- A user $U_i$ with an identity $ID_i$ sends $(pID_i, ID_i)$ to the TA.
- The TA chooses $r_i \in R \mathbb{Z}_p^*$, calculates

$$R_i = r_iP, \quad C_i = r_i + sH_0(ID_i | pID_i),$$

and sends $Cert_{ID_i} = (R_i, C_i)$ to the user.
- The user can verify the validity of the certificate by checking

$$C_iP = R_i + H_0(ID_i | pID_i) pTA.$$
Sign: A signer $U_i$ with ID$_i$ outputs a signature $(R_i, Z_i)$ of a message $m_i \in \{0, 1\}^*$ as:
- Choose $r_{2i} \in R Z^*_p$ and compute
  $$R_i = R_{1i} + r_{2i} P, \quad V_i = H_1(m_i||p_{ID}^k||ID_i)||\Delta_i),$$
  $$Z_i = r_{2i} + C_i + s_{ID} V_i.$$

Aggregate: After receiving $n$ signatures $\{(R_i, Z_i)\}_{i=1}^n$ on $\{m_i\}_{i=1}^n$, an aggregator verifies the validity of all the signatures by checking
$$Z_i P = R_i + H_0(ID_i||p_{ID}^k||TA)$$
$$+H_1(m_i||p_{ID}^k||ID_i)||\Delta_i p_{ID}^k.$$ If they hold, then the aggregator computes
$$R = \sum_{i=1}^n R_i, \quad Z = \sum_{i=1}^n Z_i$$ and returns $(R, Z)$ as an aggregate signature of $\{m_i\}_{i=1}^n$ for $(p_{ID}^k, ID_i)_{i=1}^n$.

Verification: For an aggregate signature $(R, Z)$, the end user verifies its validity by checking the following equality
$$ZP = R + \sum_{i=1}^n H_0(ID_i||p_{ID}^k||TA)$$
$$+\sum_{i=1}^n H_1(m_i||p_{ID}^k||ID_i)||\Delta_i p_{ID}^k.$$ If the equality holds, accept the aggregate signature.

III. CRYPTOANALYSIS OF FOUR CLS AND CLAS SCHEMES, AND ONE CBS SCHEME

Now, we show that the four CLS (CLAS) schemes and one CBS schemes are insecure against universal forgery attacks, type I attacks, type II attacks or malicious-but-passive-KGC attacks.

A. MALICIOUS-BUT-PASSIVE-KGC ATTACKS

In malicious-but-passive-KGC attacks, the KGC can generate the system parameters maliciously in the Setup phase to forge users’ signatures. Now, we demonstrate that Tseng et al.’s scheme and Deng et al.’s scheme are vulnerable to forge users’ signatures. Now, we demonstrate that Tseng et al.’s scheme and Deng et al.’s scheme are vulnerable to forge users’ signatures.

The First Malicious-but-passive KGC Attacks on Tseng et al.’s Scheme.

In the Setup phase, a malicious KGC first picks $\beta \in R Z^*_p$ and calculates
$$g_2 = g^\beta.$$ Then, the KGC publishes $\text{params} = (p, g_1, g_2, e, g, g_1, g_2, u^', \tilde{U}, v^', \tilde{V}, m', M, H_u, H_v, H_m)$ as the public parameters.

Then the KGC who knows $\beta$ can obtain $g_2^{2x}$ from a user public key $PK = (p k_1, p k_2) = (g^x, g_1^x)$ by computing
$$(p k_2)^{2x} = (g_1)^{2x} = (g_2)^{2x}$$ without using the user secret key $x$ associated to $PK$.

Using the value $g_2^{2x}$, the KGC can produce a signature of any message $m$ under ID as follows:
- Select $r_1, r_2, r_3 \in R Z^*_p$ and calculate $U_{ID}, U_{PK}, M$ and a signature
  $$\sigma' = (\sigma_1', \sigma_2', R_u', R_v', R_m')$$
  $$= (g_2^{2x} U_{ID} r_1 M r_2, (U_{PK}) r_3, g_1^{r_1}, g_2^{r_2}, g_3^{r_3}).$$

Then, $\sigma'_1 = (\sigma_1', \sigma_2', R_u', R_v', R_m')$ is a valid certificateless signature of $m$ under $(ID, PK)$: it passes the following verification equations
$$e(pk_1, g_1) = e(pk_2, g_2),$$
$$e(\sigma_1' \sigma_2', g) = e(g_2, p k_2^x) \cdot e(U_{ID}, R_u') \cdot e(U_{PK}, R_v') \cdot e(M, R_m').$$

This attack illustrates how the malicious KGC manipulates the system parameters to generate users’ valid signatures on any messages.

Security of Tseng et al.’s scheme relies on the intractability of computing $(g_2)^{2x}$ from $(g_1^x)$ and $g_2$ (it is the Computational Diffie-Hellman problem), but it becomes easy if one knows the discrete logarithm of $g_2$.

Thus, if the KGC chooses $\beta \in R Z^*_p$ and calculates $g_2 = g^\beta$ as the system parameter then the KGC who knows the discrete logarithm $\beta$ of $g_2$ can compute the core value $(g_2)^{2x}$ by computing $(p k_2)^x$ from its known value $g_2 = g^\beta$ without using the user secret key $r_{ID} = x$ of $PK = (p k_1, p k_2) = (g_1^x, g_2^x)$.

The KGC with the value $g_2^{2x}$ can produce valid certificateless signatures on any messages for $(ID, PK)$.

B. The Second Malicious-but-passive KGC Attacks on Tseng et al.’s CLS Scheme.

In the Setup phase, the KGC first picks $\alpha', \alpha_i, \beta', \beta_j, \delta_k \in R Z^*_p$ for $i = 1, \ldots, n_u, j = 1, \ldots, n_v$ and $k = 1, \ldots, n_m$ and computes
$$u^' = g^\alpha', \quad \tilde{U} = (g^{\alpha_i})_{i=1,...,n_u},$$
$$v^' = g^{\beta'}, \quad \tilde{V} = (g^{\beta_j})_{j=1,...,n_v},$$
$$m^' = g^{\beta'}, \quad \tilde{M} = (g^{\delta_k})_{k=1,...,n_m}.$$ Then, the KGC publishes $\text{params} = (p, e, g_1, g_2, g, g_1, g_2, u^', \tilde{U}, v^', \tilde{V}, m', M, H_u, H_v, H_m)$ as the system parameters.

The KGC gets a signature $\sigma = (\sigma_1, \sigma_2, R_u, R_v, R_m)$ of a message $m$ under $(ID, PK = (pk_1, pk_2) = (g_{1M}^x, g_{2M}^x))$ from the signing oracle, where
$$\sigma = (\sigma_1, \sigma_2, R_u, R_v, R_m) = ((d_1)^{M_{n_u}}, (d_2)^{M_{n_v}}, (d_3)^{M_{n_m}}, (d_4)^{M_{n_m}}, g_2^m).$$
Then the KGC can obtain \((g^2)^{yM}\) by computing
\[
(g^2)^{yM} = \frac{\sigma_2}{(R_q \sum_{i=1}^{n} \alpha_i \cdot (R_m)^{1/2} \sum_{i=1}^{n} \beta_i}} \cdot (g^{\sum_{i=1}^{n} \alpha_i} / (g^{\sum_{i=1}^{n} \beta_i})^{nM} = (g^2)^{yM}(U_{ID})^{yM}M^{nM} = (g^2)^{yM}(U_{ID})^{yM}M^{nM}.
\]

After getting the core value \((g^2)^{yM}\), the KGC can generate valid signatures of a new message \(m\) under \([ID, PK]\) as: pick \(r_1, r_2, r_3 \in R \mathbb{Z}_p^\ast\) and calculate \(U_{ID}, U_{PK}, M\) and a signature
\[
\begin{align*}
\sigma' &= (\sigma_1', \sigma_2', R_u', R_v', R_m') \\
&= ((g^2)^{yM}(U_{ID})^{1}, (g^2)^{yM}(U_{PK})^{2}M^{nM}, g^{\alpha}, g^{r_1}, g^{r_2}, g^{r_3}).
\end{align*}
\]
The signature is valid: it satisfies the following verification equation
\[
e(p_{k_1}, g) = e(p_{k_2}, g), \\
e(\sigma_1, \sigma_2, g) = e(g, p_{k_2}) \cdot e(U_{ID}, R_u') \cdot e(U_{PK}, R_v') \cdot e(M, R_m'),
\]
since
\[
e(\sigma_1, \sigma_2, g) = e((g^2)^{yM}(U_{ID})^{1}, (g^2)^{yM}(U_{PK})^{2}M^{nM}, g) = e(g, p_{k_2}) \cdot e(U_{ID}, R_u') \cdot e(U_{PK}, R_v') \cdot e(M, R_m').
\]

Finally, the KGC can produce valid signatures of any messages under \([PK, ID]\) without using the user secret key \(t_{ID}\) of the public key \(PK = (g^{r_1}, g^{r_2})\).

### Malicious-but-passive KGC Attacks on Deng et al.'s Scheme

- In the Setup phase, a malicious KGC first picks \(\beta \in R \mathbb{Z}_p^\ast\) and calculates
\[
Q = \beta P.
\]
Then, the KGC publishes \(\text{params} = \langle G_1, G_2, q, e, p, Q, Y, P_{pub}, H_1, H_2, H_3 \rangle\) as the public system parameters.
- The KGC who knows \(\beta\) can generate a valid certificateless signature \(\sigma_i = (U_i, V_i)\) on a message \(m_i \in \{0, 1\}^n\) under \([ID_i, PK_i]\) as:
  - Pick \(u_i \in \mathbb{Z}_q^\ast\) and calculate \(U_i = u_iP\).
  - Calculate \(h_i = H_3(m_i, U_i, ID_i, PK_i), \quad V_i = \beta(R_i + k_iP_{pub} + l_iT_i) + h_iu_iY.\)
- Then \(\sigma_i = (U_i, V_i)\) is a valid signature of \(m_i\): it passes the following verification equation
\[
e(V_i, P) = e(R_i + k_iP_{pub} + l_iT_i, Q)e(h_iU_i, Y),
\]
since
\[
e(V_i, P) = e(\beta(R_i + k_iP_{pub} + l_iT_i) + h_iu_iY, P) = e(R_i + k_iP_{pub} + l_iT_i, \beta P)e(h_iU_i, Y) = e(R_i + k_iP_{pub} + l_iT_i, Q)e(h_iU_i, Y).
\]
- The KGC can generate valid signatures of any messages under any identities.

Vulnerability of Deng et al.'s scheme against our attacks are due to the following facts:
- Security of Deng et al.'s scheme relies on the intractability of computing \(t_iQ\) from \(U_i = t_iP\) and \(Q\) (it is the Computational Diffie-Hellman problem), but it becomes easy if one knows the discrete logarithm of \(Q\).
- Thus, if the KGC chooses \(\beta \in R \mathbb{Z}_p^\ast\) and calculates \(Q = \beta P\) as the system parameter then the KGC who knows the discrete logarithm \(\beta\) of \(Q\) can compute the secret value \(t_iQ\) by computing \(\beta U_i = \beta t_iP\) from its known value \(\beta\) of \(U_i\) without the knowledge of the user secret key.
- The KGC who knows the secret value \(t_iQ\) can produce valid signatures on any messages under \([ID_i, PK_i]\).

### B. UNIVERSAL FORGERY ATTACKS

We show that in Mei et al.’s scheme, anyone without any abilities can generate valid signatures of any messages under any identities using public information.

#### Universal Forgeries on Mei et al.'s Scheme.

- Assume that an adversary \(\mathcal{A}\) can get a signature, \(sig_i = (U_i, T_i)\) of \(msg_i||t_i\) under \([vepk_i, PSUID_i, ID_{R_i}]\), where
\[
U_i = u_iP, \quad T_i = u_iJ_i + h_iE_i, \quad h_i = H_3(msg_i || t_i, vepk_i, PSUID_i, ID_{R_i}).
\]
- Then \(\mathcal{A}\) can generate a signature of a new message \(msg_i' || t_i'\) for \([vepk_i, PSUID_i, ID_{R_i}]\) as follows:
  - Compute
\[
\begin{align*}
h_i' &= H_3(msg_i' || t_i', vepk_i, PSUID_i, ID_{R_i}), \\
a_i &= h_i'/h_i.
\end{align*}
\]
  - Compute
\[
\begin{align*}
U_i' &= a_iU_i = a_iu_iP, \\
T_i' &= a_iT_i = a_iu_iJ_i + a_ih_iE_i \\
&= a_iu_iJ_i + h_i'(psk_i + vesk_i),
\end{align*}
\]
where \(E_i = psk_i + vesk_i\) and \(a_i = h_i'/h_i\). Then \(sig_i' = (U_i', T_i')\) is a valid signature on \(msg_i' || t_i'\) under \([vepk_i, PSUID_i, ID_{R_i}]\): it passes the following verification equation
\[
e(P, T_i') = e(MPK, h_i'Q_i)e(h_i'vepk_i, I_i)e(U_i', J_i),
\]
since
\[
e(P, T_i') = e(P, a_iu_iJ_i + h_i'(psk_i + vesk_i)) \\
&= e(P, a_iu_iJ_i)e(P, h_i'Q_i)e(P, h_i'vesk_i) \\
&= ee(U_i', J_i)(MPK, h_i'Q_i)e(h_i'vepk_i, I_i).
\]
- Therefore, \(\mathcal{A}\) can produce signatures on any messages for any users and RSUs, i.e. Mei et al.’s scheme is completely broken.
C. FORGERY ATTACKS ON CB-CAS

In certificate-based schemes, there exist two types of adversaries, $F_1$ and $F_2$ as:
- $F_1$ is a malicious TA with a master secret key.
- $F_2$ is a malicious signer who has its secret signing key.

The CBS scheme are similar to the CLS schemes. The adversaries $F_1$ and $F_2$ in the CBS schemes correspond to the type II adversary and the type I adversary in the CLS schemes, respectively.

In [23], the authors proved that their scheme was secure against the two types of adversaries, $F_1$ and $F_2$ under the intractability assumption of the Elliptic Curve Discrete Logarithm problem. Here, we show that CB-CAS is insecure against the two types of adversaries, $F_1$ and $F_2$ in spite of their security proofs. We also show that, in the scheme, a user can generate valid certificates of the other users from its own certificate received by the TA.

**The First Attack on CB-CAS (Type II Attack).**

- Let $F_1$ be a malicious TA with the master secret key $s$.
- Assume that $F_1$ intends to generate a valid signature of a message under $ID_i$. Since $F_1$ can access the signing oracle, it gets a valid signature $(R_i, Z_i)$ of the message $m_i$ for the user $ID_i$, where

$$
R_i = R_{i1} + R_{i2}P,
V_i = H_1(m_i||p_{ID_i}^k||ID_i||)\Delta,
Z_i = r_{i1} + s_{ID_i}V_i,
$$

where $R_i = r_{i1}P$ and $C_i = r_{i1} + s_{ID_i}V_i$.

- First, $F_1$ with the master secret key $s$ can calculate

$$
V_i = H_1(m_i||p_{ID_i}^k||ID_i||)\Delta,
Z_i - s_{ID_i}V_i = r_{i1} + s_{ID_i}V_i,
V_i^{-1}[Z_i - s_{ID_i}V_i] = V_i^{-1}(r_{i1} + s_{ID_i}V_i),
$$

- Let $X = V_i^{-1}(r_{i1} + s_{ID_i}V_i)$, then, using the two values, $X$ and $s$, $F_1$ can generate a valid signature $(R'_i, Z'_i)$ of a new message $m'_i$ under $ID_i$, $PK_i$ as

$$
V'_i = H_1(m'_i||p_{ID_i}^k||ID_i||)\Delta,
R'_i = V'_i \cdot V_i^{-1} \cdot R_i = V'_i \cdot V^{-1}(r_{i1} + r_{i2})P,
Z'_i = X \cdot V'_i + s_{ID_i}(V_i)||p_{ID_i}^k|| = [V_i^{-1}(r_{i1} + r_{i2})] + s_{ID_i}V'_i + s_{ID_i}(V_i)||p_{ID_i}^k||.
$$

Then $(R'_i, Z'_i)$ is valid: it satisfies the following verification equation

$$
Z'_iP = R_i + H_0(ID_i|||p_{ID_i}^k||p_{TA}^k)P + H_1(m_i||p_{ID_i}^k||ID_i||)\Delta p_{ID_i}^k,
$$

since

$$
Z'_iP = [V_i^{-1}(r_{i1} + r_{i2}) + s_{ID_i}]V'_i + s_{ID_i}(V_i)||p_{ID_i}^k||P = R'_i + V'_ip_{ID_i}^k + H_0(ID_i|||p_{ID_i}^k||p_{TA}^k),
$$

$$
= R'_i + H_1(m'_i||p_{ID_i}^k||ID_i||)\Delta p_{ID_i}^k + H_0(ID_i|||p_{ID_i}^k||p_{TA}^k),
$$

- Finally, $F_1$ forges successfully a valid signature of any message $m'_i$ ($\neq m_i$) for $ID_i$ without using the user secret key $s_{ID_i}$ corresponding to $ID_i$. Therefore, $F_1$ can produce valid signatures of any messages.

**The Second Attack on Verma et al.'s Scheme (Type I Attack).**

- Let $F_2$ be a malicious signer who has the secret signing key $s_{ID_i}$ for $ID_i$.
- Assume that $F_2$ wants to generate a valid signature of a message under $ID_i$. Since $F_2$ can access the signing oracle, it gets a valid signature $(R_i, Z_i)$ of the message $m_i$ for $ID_i$, where

$$
R_i = R_{i1} + r_{i2}P,
V_i = H_1(m_i||p_{ID_i}^k||ID_i||)\Delta,
Z_i = r_{i2} + C_i + s_{ID_i}V_i.
$$

- First, $F_2$ who knows $s_{ID_i}$ can compute

$$
V_i = H_1(m_i||p_{ID_i}^k||ID_i||)\Delta,
Z_i - s_{ID_i}V_i = r_{i2} + C_i.
$$

Let $Y = r_{i2} + C_i$. Then, using the two values, $Y$ and $s_{ID_i}$, $F_2$ can generate a signature $(R'_i, Z'_i)$ of a new message $m'_i$ for $ID_i$, $PK_i$ as

$$
V'_i = H_1(m'_i||p_{ID_i}^k||ID_i||)\Delta,
R'_i = R_i,
Z'_i = Y + s_{ID_i}V'_i = r_{i2} + C_i + s_{ID_i}V'_i.
$$

Then $(R'_i, Z'_i)$ is a valid signature of $m'_i$ under $ID_i$: it passes the following verification equation

$$
Z'_iP = [r_{i2} + C_i + s_{ID_i}V'_i]P = [r_{i2} + r_{i1} + s_{ID_i}(V_i)||p_{ID_i}^k||] + s_{ID_i}V'_iP = R'_i + H_0(ID_i|||p_{ID_i}^k||p_{TA}^k) + H_1(m'_i||p_{ID_i}^k||ID_i||)\Delta p_{ID_i}^k,
$$

where $C_i = r_{i1} + s_{ID_i}(V_i)||p_{ID_i}^k||$.

- Finally, $F_2$ succeeds in forging signatures of any messages $m'_i$ ($\neq m_i$) for $ID_i$ without the knowledge of the master secret key $s$. Therefore, $F_2$ can produce valid signatures on any messages under $ID_i$.

**Certificate Forgery Attacks.**

- Suppose a user $U_i$ with $ID_i$ received its own certificate $Cert_{ID_i} = (R_{i1}, C_i)$ from the TA, where

$$
R_{i1} = R_{i1}P, C_i = r_{i1} + s_{ID_i}(V_i)||p_{ID_i}^k||,
$$

- Then $U_i$ can generate valid certificates of other users $U_j$ with $ID_j$, $p_{ID_j}^k$ for $j \neq i$:
  - Let $H_0(ID_i|||p_{ID_i}^k||p_{TA}^k) - H_0(ID_i|||p_{ID_j}^k||p_{TA}^k) = \alpha$. Then $U_i$ can compute $U_j$'s certificate as

$$
R_{ij} = R_{i1} + \alpha \cdot p_{TA}^k, C_j = C_i.
$$
- Then, $Cert_{ID} = (R_{i1}, C_i)$ is a valid $U_i$’s certificate since its passes the following equation

$$C_jP = R_{ij} + H_0(ID_j)||p^k_{ID_j})p^k_{TA},$$

where

$$C_jP = C_jP = \left[r_{ij} + sH_0(ID_j)||p^k_{ID_j})\right]P$$

$$= (r_{ij} + s(H_0(ID_j)||p^k_{ID_j}) + \alpha)P$$

$$= (r_{ij} + \alpha + sH_0(ID_j)||p^k_{ID_j})P$$

$$= R_{ij} + H_0(ID_j)||p^k_{ID_j})p^k_{TA}.$$

Thus, the user $U_i$ can generate other users’ certificates $Cert_{ID}$ from its certificate $Cert_{ID}$ for $j \neq i$.

If we set the user as the second type adversary who knows users’ secret keys then the user using other users’ certificates can produce any signatures on any messages for the users.

### IV. HOW TO DESIGN SECURE CLS AND CLAS SCHEMES

In the previous section, we presented the several algebraic attacks on the four CLS and CLAS schemes, and one CBS scheme although security of all the schemes were proved under the intractability assumptions of the mathematical problems. Our attacks do not mean that a random instance of each hard problem is solved, but all the security proofs against the type I adversary or the type II adversary are flawed neglecting the various attacks due to the algebraic relations of the underlying group elements. Here, we provide design principles for the construction of secure CLS and CLAS schemes to prevent various algebraic attacks including our attacks.

1. **Integrity of Random Values.** If one uses randomized probabilistic algorithms for the partial private key extract algorithm and the signing algorithm instead of deterministic algorithms (for example, BLS signature scheme [25]) then random values should be included in the input of a hash function to prevent forgery attacks mounted by modifying the random values. It guarantees the integrity of the random values.

   - In particular, as in Mei et al.’s scheme, to design compact CLAS schemes whose signature lengths are constant, some CLAS schemes remove the random values from the input of the hash functions that causes the universal forgery attacks, the type I attacks and the type II attacks. Vulnerabilities of the schemes [16], [23] against our attacks are due that it does not use appropriate hash values including the random values in the partial private key extraction and signing for compactness.

   - In Mei et al.’s CLAS scheme, if the hash value

   $$h_i = H_3(msg_i||t_i, vepk_i, PSUID_i, ID_{R_i})$$

   is replaced with

   $$h_i = H_3(msg_i||t_i, vepk_i, PSUID_j, ID_{R_i}, U_i),$$

   where $U_i = u_iP$ is a random value generated in signing algorithm then our attack can be prevented.

   - In CertExtract of CB-CAS, a user $U_i$’s certificate $Cert_{ID} = (R_{i1}, C_i)$ is generated as

   $$R_{i1} = r_{i1}P, C_i = r_{i1} + sH_0(ID_i)||p^k_{ID_i}).$$

   The authors in [23] removed the value $R_{i1}$ from in the input of $H_0(ID_i)||p^k_{ID_i}$ not to include $R_{i1}$ in the signatures for compactness. Our certificate forgery attack allows the user to generate the other users’ certificates from its own certificate by modifying the value of $R_{i1}$ to satisfy the certificate verification equation. To prevent the attack, the hash value $H_0(ID_i)||p^k_{ID_i}$ should be replaced with $H_0(ID_i)||p^k_{ID_j}||R_{i1})$. Then, the user cannot modify $R_{i1}$ as he intended, since $R_{i1}$ and $H_0(ID_i)||p^k_{ID_i}||R_{i1}$ should be computed at the same time. Thus, the random value $R_i = r_{i2}P + R_{i1}$ related to $R_{i1}$ should be separated and $R_{i1}$ should be included in the signature.

   - To prevent the first attack on CB-CAS, the random values, $R_i$ and $R_{i1}$, should be included in the input of the hash function $H_1$ as

   $$V_i = H_1(m||p^k_{ID_i}||ID_i||\Delta||R_{i1}||R_i).$$

   and $R_{i1}$ should be also included in the input of $H_0$ as

   $$H_0(ID_i)||p^k_{ID_i}||R_{i1}).$$

   These improvements can prevent the attacks by modifying $R_i$ or $R_{i1}$ so that the forged signatures satisfy the verification equations.

   - To prevent the second attack on CB-CAS, a new hash value $H_0(m||p^k_{ID_i}||ID_i||\Delta||R_{i1}||R_i)$ should be multiplied by $r_{i2}$ or $C_i$.

   - These improvements can prevent the attacks by using the information obtained from the previous signature of a message $m$ to forge a signature of a new message $m’$. To design the compact CB-AS scheme, Verma et al. eliminated the random values, $R_{i1}$ and $R_i$, from the inputs of all the hash values which caused the vulnerabilities of their scheme against our attacks. Thus, any random values in the schemes should be guaranteed their integrity to prevent the forgery attacks by modifying the random values in partial private keys or signatures.

   The inclusion of the random values in the input of hash function can prevent the attacks, but makes the construction of compact CLAS schemes and compact CB-AS schemes impossible since the random values cannot be aggregated into a single value. It is difficult to design a secure compact AS scheme since the generations of users’ certificates and random values in signatures should be contained in appropriate hash values. Therefore, it still remains an open problem to construct a secure compact AS scheme with the constant-size aggregate signatures.

2. **Binding Identities, Public Keys, Random Values with Messages.** In signing algorithms, it needs to bind a user identity, a user public key, random values generated in the partial private key extract algorithm and the signing algorithm with a message being signed. It can prevent various forgery
attacks by using previous signatures since it guarantees their connectivity to the signatures.

- The cause of the type I attack on Tseng et al.’s scheme is that there is no binding between a message being signed and a user’s identity or a user’s public key. The attack can be prevented by adding information about the user public key $PK$ and its corresponding identity $ID$ to the hash function $H_m$ as

$$m = H_m(m, PK, ID)$$

for a message $m$, adding the user public key $PK$ as

$$u = H_u(ID, PK)$$

and adding the identity $ID$ as

$$v = H_v(PK, ID).$$

Then the type I adversary cannot use the value

$$\sigma_2 = (g_2)^{v \cdot (U_{PK})^x} M^{n_m}$$

for $\{ID, PK^+\}$ to forge a signature of the message under $\{ID^+, PK^+\}$ since the value $U_{PK}$ is computed from $v = H_v(PK^+, ID)$, so it is not matched to the value from $\{ID, PK\}$. In this case, the hash value $m = H_m(m, PK, ID)$ plays a role in binding the message $m$ with $\{ID, PK\}$ and the identity with the user public key, respectively.

Thus, the CLS schemes should be designed so that the previous signature on a message must not be used to forge signatures other messages.

3. Type I Attacks: The exposure of the user secret key does not lead to the exposure of the master key or its related information. To prevent the type I attacks, CLS schemes should be designed so that the user public key cannot be modified to remove the value associated to the master public key from the verification equation. Using algebraic relations in the group, if one can find an appropriate user public key that makes the master public key removable in the verification equation then the type I attack is successful: its forgery is possible without using the partial private keys.

- In the second attack on CB-CAS, some algebraic relations in the group make the adversary get a target value associated to the master secret by subtracting the user secret key from the previous signatures.

Thus, the CLS scheme should be designed so that the useful information related to the user secret key cannot be extracted from the replaced public keys and their arithmetics since an adversary with the useful information can forge users’ signatures without using the users’ secret keys.

4. Type II Attacks: The exposure of the master secret key does not lead to the exposure of the user secret key or its related information. To prevent type II attacks, CLS schemes should be designed so that the KGC with the master secret cannot compute the values associated to a user secret key from the signatures obtained by the signing oracle. As seen in the first attack on CB-CAS, even though the KGC does not get the user secret key itself, the attacks succeed in forging signatures from the values associated to user secret key.

- The first attack on CB-CAS, algebraic relations in the group make the attacker get the target values associated to the user secret key by subtracting the value related to the master secret from the previous signatures.

Thus, the CLS scheme should be designed so that the useful information associated to the user secret key cannot be extracted from the master secret and their arithmetics.

5. Malicious-but-Passive Attacks: Destroy algebraic structure of the group elements in the system parameters. To prevent the malicious-but-passive KGC attacks, the algebraic structure of every group element in system parameters except the generators should be destroyed. It can be possible by using the hash functions.

- In Tseng et al.’s scheme, the algebraic structure of $g_2, \hat{m}, \hat{u}, \hat{v}, \hat{\nu}$ and $\hat{M}$ except the generator $g$ and the master public key $g_1$ should be hidden by the hash function to prevent the attacks. The first malicious-but-passive-KGC attack on Tseng et al.’s scheme is due that the KGC knows the discrete logarithm of $g_2$ by computing $g_2 = g^\beta$ for its own choice $\beta$. If we set $g_2$ as the output of the hash function,

$$g_2 = \hat{H}(g) \in G_1,$$

then our attack can be prevented since the KGC cannot compute the discrete logarithm of $\hat{H}(g)$ due to the hardness of the Discrete Logarithm problem in $G_1$. The second malicious-but-passive KGC attacks on the scheme can be prevented if the vectors $m', \hat{M}, u', \hat{U}, v', \hat{\nu}$ in the system parameters are computed as the hash values of randomly chosen vectors as

$$m' = \hat{H}(\hat{m}), \quad u' = \hat{H}(\hat{u}), \quad v' = \hat{H}(\hat{v}),$$

$$\hat{U} = (\hat{u}_i = \hat{H}(\hat{u}_i))_{i=1, \ldots, n_u},$$

$$\hat{\nu} = (\hat{v}_i = \hat{H}(\hat{\nu}_i))_{i=1, \ldots, n_v},$$

$$\hat{M} = (\hat{m}_i = \hat{H}(\hat{m}_i))_{i=1, \ldots, n_m}.$$
the victim. Although the underlying mathematical problems are hard, the KGC’s malicious behaviors make the specific cases of the problems resolved. Thus, secure CLS (CLAS) schemes should be secure against these malicious-but-passive KGC attacks.

V. CONCLUSION
Due to the suitability of CLS and CLAS schemes for IoT environments, similar CLS and CLAS schemes have been proposed for a long time and, despite their security proofs, they have been attacked and modified to prevent the attacks. Even now, similar design methods and similar attacks on the schemes are being repeated. We first showed that recent four CLS and CLAS schemes, and one CB-signature were insecure against the universal forgery attacks, the type I attacks, the type II attacks or the malicious-but-passive KGC attacks. We discuss their security flaws, causes and countermeasures. We then presented design principles to prevent various algebraic attacks including our attacks. The design principles will help in the construction of secure CLS and CLAS schemes against the previous and potential attacks. We focused on security analysis of the type I attacks, the type II attacks, the universal forger attacks and the malicious KGC. However, in the multi-user setting consisting of the KGC and multiple users, insider attacks and colluding attacks should be considered. Future research will include security against insider attacks and colluding attacks in the multi-user setting.

REFERENCES

[1] S. S. Al-Riyami and K. G. Paterson, “Certificateless public key cryptography,” in Asiacrypt, vol. 2894. Berlin, Germany: Springer, 2003, pp. 452–473.

[2] W. Diffie and M. Hellman, “New directions in cryptography,” IEEE Trans. Inf. Theory, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[3] A. Shamir, “Identity-based cryptosystems and signature schemes,” in Advances in Cryptology—CRYPTO (Lecture Notes in Computer Science), vol. 196, G. R. Blakley and D. Chaum, Eds. Heidelberg, Germany: Springer-Verlag, 1984, pp. 47–53.

[4] S. Hussain, S. S. Ullah, I. Ali, J. Xie, and V. N. Irukulla, “Certificateless signature schemes in industrial Internet of Things: A comparative survey,” Comput. Commun., vol. 181, pp. 116–131, Jan. 2022.

[5] M. A. Khan, I. Ullah, M. H. Alsharif, A. H. Alghthani, A. A. Aly, and C.-M. Chen, “An efficient certificate-based aggregate signature scheme for Internet of Drones,” Secur. Commun. Netw., vol. 2022, pp. 1–9, Jan. 2022.

[6] M. H. Au, Y. Mu, J. Chen, D. S. Wong, J. K. Liu, and G. Yang, “Malicious KGC attacks in certificateless cryptography,” in Proc. 2nd ACM Symp. Inf., Comput. Commun. Secur., Mar. 2007, pp. 302–311.

[7] R. Canetti, O. Goldreich, and S. Halevi, “The random Oracle methodology, revisited (preliminary version),” in Proc. 13th Annu. ACM Symp. Theory Comput. (STOC), 1998, pp. 209–218.

[8] M. Bellare, A. Boldyreva, and A. Palacio, “An uninstantiable random-oracle-model scheme for a hybrid-encryption problem,” in Advances in Cryptology EUROCRYPT 2004 (Lecture Notes in Computer Science), vol. 3027, Cham, Switzerland: Springer, 2004, pp. 171–188.

[9] K. G. Paterson and J. C. N. Schuldt, “Efficient identity-based signatures secure in the standard model,” in Information Security and Privacy (Lecture Notes in Computer Science), vol. 4058. Cham, Switzerland: Springer, 2006, pp. 207–222.

[10] K.-H. Yeh, “A secure transaction scheme with certificateless cryptographic primitives for IoT-based mobile payments,” IEEE Syst. J., vol. 12, no. 2, pp. 2027–2038, Jun. 2018.

[11] M. E. S. Saeed, Q.-Y. Liu, G. Tian, B. Gao, and F. Li, “Remote authentication schemes for wireless body area networks based on the Internet of Things,” IEEE Internet Things J., vol. 5, no. 6, pp. 4926–4944, Dec. 2018.

[12] A. Karati, C.-A. Fan, and R.-H. Hsu, “Provably secure and generalized signcryption with public verifiability for secure data transmission between resource-constrained IoT devices,” IEEE Internet Things J., vol. 6, no. 6, pp. 10431–10440, Dec. 2019.

[13] J. Liu, L. Wang, and Y. Yu, “Improved security of a pairing-free certificateless aggregate signature in healthcare wireless medical sensor networks,” IEEE Internet Things J., vol. 7, no. 6, pp. 5256–5266, Jun. 2020.

[14] N. B. Gayathri, G. Thumbar, P. R. Kumar, M. Z. U. Rahman, P. V. Reddy, and A. Lay-Ekaukkile, “Efficient and secure pairing-free certificateless aggregate signature scheme for healthcare wireless medical sensor networks,” IEEE Internet Things J., vol. 6, no. 5, pp. 9064–9075, Oct. 2019.

[15] G. Thumbar, G. S. Rao, P. V. Reddy, N. B. Gayathri, D. V. R. K. Reddy, and M. Padnavathamma, “Efficient and secure certificateless aggregate signature-based authentication scheme for vehicular ad hoc networks,” IEEE Internet Things J., vol. 8, no. 3, pp. 1908–1920, Feb. 2021.

[16] Q. Mei, H. Xiong, J. Chen, M. Yang, S. Kumari, and M. K. Khan, “Efficient certificateless aggregate signature with conditional privacy preservation in IoV,” IEEE Syst. J., vol. 15, no. 1, pp. 245–256, Mar. 2021.

[17] K.-A. Shim, “Cryptanalysis of two signature schemes for IoT-based mobile payments and healthcare wireless medical sensor networks,” IEEE Access, vol. 8, pp. 167203–167208, 2020.

[18] K. A. Shim, “Universal forgery attacks on remote authentication schemes for wireless body area networks based on Internet of Things,” IEEE Internet Things J., vol. 6, no. 5, pp. 5973–5984, Oct. 2019.

[19] Y. Liao, Y. Liu, Y. Liang, Y. Wu, and X. Nie, “Revisit of certificateless signature scheme used to remote authentication schemes for wireless body area networks,” IEEE Internet Things J., vol. 7, no. 3, pp. 2160–2168, Mar. 2020.

[20] L. Deng, B. Ning, and Y. Jiang, “A lightweight certificateless aggregation signature scheme with provably security in the standard model,” IEEE Syst. J., vol. 14, no. 3, pp. 4242–4251, Sep. 2020.

[21] Y.-F. Tseng, C.-I. Fan, and C.-W. Chen, “Top-level secure certificateless signature scheme in the standard model,” IEEE Syst. J., vol. 13, no. 3, pp. 2763–2774, Sep. 2019.

[22] C. Gentry, “Certificate-based encryption and the certificate revocation problem,” in Proc. Int. Conf. Theory Appl. Cryptograph. Techn., 2003, pp. 272–293.

[23] G. K. Verma, B. B. Singh, N. Kumar, O. Kaiwartya, and M. S. Obaidat, “PPFCBAS: Pairing free and provable certificate-based aggregate signature scheme for the e-healthcare monitoring system,” IEEE Syst. J., vol. 14, no. 2, pp. 1704–1715, Jun. 2020.

[24] M. Girault, “Self-certified public keys,” in Proc. 10th Annu. Int. Conf. Theory Appl. Cryptogr. Techn., 1991, pp. 490–497.

[25] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, “Aggregate and verifiably encrypted signatures from bilinear maps,” in Advances in Cryptology EUROCRYPT 2003 (Lecture Notes in Computer Science), vol. 3027. Cham, Switzerland: Springer, 2003, pp. 416–432.