Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 1908632, 9 pages
https://doi.org/10.1155/2018/1908632

Research Article

Cryptanalysis and Improvement of Three Certificateless Aggregate Signature Schemes

Xiaodong Yang,[1,2] **Yutong Li**,[1] **Chunlin Chen**,[1] **Likun Xiao**,[1] **and Caifen Wang**[1]

[1]*College of Computer Science and Engineering, Northwest Normal University, Lanzhou, Gansu 730070, China*
[2]*State Key Laboratory of Cryptology, Beijing 100878, China*

Correspondence should be addressed to Xiaodong Yang; y200888@163.com

Received 14 March 2018; Revised 13 July 2018; Accepted 24 July 2018; Published 23 August 2018

Academic Editor: Nazrul Islam

The certificateless aggregate signature (CLAS) scheme is a very important data aggregation technique that compresses a large number of signatures from different users into a short signature. CLAS can reduce the total length of a signature and the computational overhead of signature verification and is therefore highly suitable for resource-constrained network environments. Many CLAS schemes have been proposed in recent years, but the construction of a secure and efficient CLAS scheme remains important. In 2018, Li et al. found that the CLAS scheme proposed by He et al. could not resist malicious-but-passive KGC attacks, and they presented an improved CLAS scheme. Du et al. proposed a CLAS scheme with the constant aggregate signature length and claimed that their scheme was resistant to forgery attacks. Chen et al. designed a CLAS scheme with efficient verification and proved that their CLAS scheme was secure in the random oracle model. In this paper, we demonstrate that Li et al.'s CLAS scheme, Du et al.'s CLAS scheme, and Chen et al.'s CLAS scheme are insecure against coalition attacks and present concrete examples. That is, an attacker can forge a valid aggregate signature using some illegal single signatures. To withstand suck attacks, we propose an improved CLAS scheme based on Chen et al.'s CLAS scheme.

1. Introduction

The traditional signature scheme provides security services such as integrity of the message, nonrepudiation of the signer, and user authentication. However, the scheme relies on the public key infrastructure (PKI) and needs to store and verify many certificates [1]. To solve this problem, Shamir [2] proposed the concept of identity-based signature (IBS), where the user's public key is replaced with an email address or other unique identity information of the user, and a fully trusted key generation center (KGC) produces the user's private key. Nevertheless, IBS suffers from the key escrow problem since the KGC knows the private keys of all users and can do anything on behalf of the user without being detected. To eliminate the use of certificates and to avoid the key escrow problem, certificateless signature (CLS) was introduced by Al-Riyami and Paterson [3]. In CLS, the user's public key is self-generated, and the user's private key consists of two parts: a secret value selected by the user and a partial private key generated by a semitrusted KGC. Consequently, CLS can avoid the security flaws of IBS without requiring certificates. Although some efficient CLS schemes have been presented [4–8], more secure CLS schemes are desired for practical applications.

An aggregate signature scheme [9] can aggregate multiple signatures of different messages into a short signature, and the verifier is able to determine the validity of each signature participating in the aggregation by verifying the validity of the aggregate signature. Aggregate signatures reduce storage and bandwidth overhead and are thus highly useful in wireless communication environments. Because of the advantages of CLS and aggregate signatures, some researchers have focused on certificateless aggregate signature (CLAS) schemes [10–12].

In 2013, Xiong et al. [13] presented an efficient CLS scheme and used it to design a CLAS scheme with constant pairing computations. However, He et al. [14] found that Xiong et al.'s CLS and CLAS schemes [13] were not secure against malicious KGC attacks, and they proposed an improved CLAS scheme. Very recently, Li et al. [15] showed that He et al.'s CLAS scheme [14] still could not

resist malicious-but-passive KGC attacks. Later, Li et al. [15] designed an improved CLAS scheme and claimed that their CLAS scheme was resistant to forgery attacks. Ming et al. [16] gave a CLAS scheme in which the length of aggregate signature is constant. However, Du et al. [17] pointed out that Ming et al.'s CLAS scheme [16] could not resist malicious KGC attacks and proposed a new CLAS scheme. Chen et al. [18] presented a CLAS scheme with efficient verification, but their CLAS scheme was shown to be insecure against Type I and Type II adversaries [19–21].

However, most CLAS schemes do not consider coalition attacks [22], which are a type of practical and powerful attack in which an attacker attempts to generate a valid aggregate signature by using illegal individual signatures. Once such an attack is successful, the validity of an aggregate signature cannot guarantee that each individual signature participating in the aggregation is valid. This requires that a secure CLAS scheme should be able to resist coalition attacks. Unfortunately, we note that Li et al.'s CLAS scheme [15], Du et al.'s CLAS scheme [17], and Chen et al.'s CLAS scheme [18] are insecure against coalition attacks since the attacker can forge a valid aggregate signature through illegal single signatures generated by conspiracy from two or more internal signers.

1.1. Contributions. In this paper, we provide a cryptanalysis of three CLAS schemes. We first present an attack against Li et al.'s CLAS scheme [15] to show that their CLAS scheme is insecure under practical coalition attacks. Then, we point out that Du et al.'s CLAS scheme [17] cannot resist coalition attacks by giving a concrete attack. Furthermore, we demonstrate the security weakness of Chen et al.'s CLAS scheme [18] by presenting a forgery attack from a coalition of insider signers. Based on Chen et al.'s CLAS scheme, we construct a new CLAS scheme that is secure against coalition attacks.

1.2. Paper Organization. The remainder of this paper is organized as follows. Section 2 reviews some preliminaries. Section 3 analyses the security of Li et al.'s CLAS scheme. Section 4 gives cryptanalysis of Du et al.'s CLAS scheme. Section 5 analyses the security of Chen et al.'s CLAS scheme. Section 6 presents an improved CLAS scheme. Finally, Section 7 concludes the paper.

2. Preliminaries

In this section, we briefly review bilinear pairing and the definitions of CLS and CLAS.

2.1. Bilinear Pairing. Suppose that G_1 and G_2 are two multiplicative cyclic groups of prime order p and that g is a generator of G_1. An efficiently computable map $e : G_1 \times G_1 \longrightarrow G_2$ is said to be a bilinear pairing if it satisfies the following conditions [18]:

(i) *Bilinearity*: $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$ for any $a, b \in Z_p$.

(ii) *Nondegeneracy*: $e(g, g) \neq 1$.

2.2. Definition of Certificateless Signature. A CLS scheme is defined by the following five algorithms:

(i) **Setup**: Upon input of a security parameter $\lambda \in Z$, this algorithm outputs the public parameters pp and the PKG's master secret key msk.

(ii) **PartialKeyGen**: Upon input of pp, msk, and a user identity ID_i, this algorithm outputs a partial private key psk_i corresponding to ID_i.

(iii) **UserKeyGen**: Upon input of pp and psk_i, this algorithm outputs ID_i's secret value usk_i, public key upk_i, and private key $sk_i = (psk_i, usk_i)$.

(iv) **Sign**: Upon input of pp, an identity ID_i's private key sk_i, and a message m_i, this algorithm outputs a signature σ_i on m_i.

(v) **Verify**: Given an identity ID_i, a public key upk_i, a message m_i, and a signature σ_i, a verifier accepts σ_i if σ_i is a valid signature on m_i with respect to ID_i and upk_i; otherwise, the verifier rejects σ_i.

In general, two types of attackers exist in a CLS scheme, i.e., Type I and Type II adversaries [3]. A Type I adversary models an outside attacker who is able to determine the user's secret value or replace the user's public key at will but who does not have access to the KGC's master secret key and the user's partial private key. A Type II adversary models a malicious KGC who knows the KGC's master secret key and generates the user's partial private key but who is unable to access the user's secret value or replace the user's public key. In the original security model for a CLS scheme [3], a Type II adversary represents an honest-but-curious KGC who always honestly generates the system parameters and the user's partial private key according to the specifications of the CLS scheme. In the stronger security model proposed by Au et al. [23], a Type II adversary represents a malicious-but-passive KGC who is dishonest from the beginning of the system setup. The malicious-but-passive KGC adversary sets some trapdoors in the system parameters and the master secret key and then uses these trapdoors to launch malicious attacks to compromise the security of the CLS scheme. However, some existing CLS schemes [6–8] have been shown to be insecure against malicious-but-passive KGC attacks.

2.3. Definition of Certificateless Aggregate Signature. A CLAS scheme is defined by the following seven algorithms:

(i) The **Setup**, **PartialKeyGen**, **UserKeyGen**, **Sign**, and **Verify** algorithms are the same as those in CLS described in Section 2.

(ii) **Aggregate**: Upon input of n message-signature pairs $\{(m_1, \sigma_1), \ldots, (m_n, \sigma_n)\}$ from n users, this algorithm outputs an aggregate signature σ on messages $\{m_1, \ldots, m_n\}$.

(iii) **AggregateVerify**: Given n identities $\{ID_1, \ldots, ID_n\}$, n public keys $\{upk_1, \ldots, upk_n\}$, and an aggregate signature σ on n messages $\{m_1, \ldots, m_n\}$, a verifier accepts σ if σ is valid; otherwise, the verifier rejects σ.

The security model for the CLAS scheme is almost the same as the security model for the CLS scheme, except that an adversary is unable to forge a valid aggregate signature if at least one of the signatures involved in the aggregation is invalid. To save space, we omit the detailed description. Please refer to [1, 3, 23] for the security model of the CLS scheme or the CLAS scheme.

3. Cryptanalysis of Li et al.'s CLAS Scheme

3.1. Review of Li et al.'s Scheme. In 2018, Li et al. [15] presented a CLAS scheme that consists of the following seven algorithms:

(i) **Setup**: Given a security parameter $\lambda \in Z$, the KGC generates the public parameters pp and the master secret key msk by performing the following steps:

(a) Choose two cyclic groups G_1 and G_2 of prime order p, a generator g of G_1 and a bilinear pairing $e : G_1 \times G_1 \longrightarrow G_2$.

(b) Select a random integer $s \in Z_p^*$ and compute $P_{pub} = g^s$.

(c) Pick four cryptographic hash functions $H_1, H_4 : \{0,1\}^* \longrightarrow G_1$ and $H_2, H_3 : \{0,1\}^* \longrightarrow Z_p^*$.

(d) Keep the master secret key $msk = s$ secret and publish public parameters $pp = \{G_1, G_2, p, g, , e, P_{pub}, H_1, H_2, H_3, H_4\}$.

(ii) **PartialKeyGen**: Given a user's identity ID_i, the KGC computes $Q_i = H_1(ID_i)$ and $psk_i = (Q_i)^s$. Then, the KGC sends ID_i's partial private key psk_i to the user.

(iii) **UserKeyGen**: The user with identity ID_i generates a public/private key pair (upk_i, sk_i) according to the following steps:

(a) Select a random integer $x_i \in Z_p^*$ and compute the public key $upk_i = g^{x_i}$.

(b) Set the secret value $usk_i = x_i$ and the private key $sk_i = (psk_i, usk_i) = ((Q_i)^s, x_i)$.

(iv) **Sign**: To sign a message m_i, a signer with identity ID_i performs the following:

(a) Select a random integer $r_i \in Z_p^*$ and compute $R_i = g^{r_i}$.

(b) Compute $h_i = H_2(0, m_i, ID_i, upk_i, R_i)$, $k_i = H_3(1, m_i, ID_i, upk_i, R_i)$, $Q = H_4(p, g, P_{pub})$ and $Y_i = (psk_i)^{h_i} \cdot (P_{pub})^{r_i} \cdot Q^{k_i x_i} \cdot Q^{r_i}$.

(c) Output a signature $\sigma_i = (R_i, Y_i)$ on m_i.

(v) **Verify**: Given an identity ID_i, a public key upk_i, a message m_i, and a signature $\sigma_i = (R_i, Y_i)$, a verifier first computes $Q_i = H_1(ID_i)$, $h_i = H_2(0, m_i, ID_i, upk_i, R_i)$, $k_i = H_3(1, m_i, ID_i, upk_i, R_i)$ and $Q = H_4(p, g, P_{pub})$. Then, the verifier checks the following equation:

$$e(Y_i, g) = e\left(R_i (Q_i)^{h_i}, P_{pub}\right) e\left((upk_i)^{k_i} R_i, Q\right). \quad (1)$$

If the above equation holds, then the verifier accepts σ_i; otherwise, the verifier rejects σ_i.

(vi) **Aggregate**: Given n message-signature pairs $\{(m_1, \sigma_1 = (R_1, Y_1)), \ldots, (m_n, \sigma_n = (R_n, Y_n))\}$ from n users, the aggregator calculates an aggregate signature $\sigma = (R_1, \ldots, R_n, Y)$ on $\{m_1, \ldots, m_n\}$, where $Y = \prod_{i=1}^{n} Y_i$.

(vii) **AggregateVerify**: Given n identities $\{ID_1, \ldots, ID_n\}$, n public keys $\{upk_1, \ldots, upk_n\}$, and an aggregate signature $\sigma = (R_1, \ldots, R_n, Y)$ on n messages $\{m_1, \ldots, m_n\}$, a verifier first computes $Q_i = H_1(ID_i)$, $h_i = H_2(0, m_i, ID_i, upk_i, R_i)$, $k_i = H_3(1, m_i, ID_i, upk_i, R_i)$ and $Q = H_4(p, g, P_{pub})$ for $1 \le i \le n$. Then, the verifier checks the following equation:

$$e(Y, g) = e\left(\prod_{i=1}^{n} R_i (Q_i)^{h_i}, P_{pub}\right) e\left(\prod_{i=1}^{n} (upk_i)^{k_i} R_i, Q\right). \quad (2)$$

If this equation holds, then the verifier accepts σ; otherwise, the verifier rejects σ.

3.2. Attack on Li et al.'s Scheme. Li et al. [15] proved that their CLAS scheme is existentially unforgeable in the random oracle model. In this subsection, we show that Li et al.'s CLAS scheme [15] is insecure against coalition attacks from internal signers. That is, two or more internal signers who generate multiple illegal single signatures can collude to generate a valid aggregate signature. Without loss of generality, suppose that $\mathcal{U}_1$ and $\mathcal{U}_2$ are two internal signers. Let ID_1 and upk_1 be the identity and public key of $\mathcal{U}_1$, respectively. ID_2 and upk_2 are the identity and public key of $\mathcal{U}_2$, respectively. $\mathcal{U}_1$ generates an invalid signature σ_1^* of message m_1, and $\mathcal{U}_2$ generates an invalid signature σ_2^* of message m_2. Then, $\mathcal{U}_1$ colludes with $\mathcal{U}_2$ to generate a valid aggregate signature σ^* for $\{m_1, m_2\}$. The detailed attack process is described as follows.

(1) $\mathcal{U}_1$ randomly selects $r_1 \in Z_p^*$ and calculates $R_1^* = g^{r_1}$, $h_1 = H_2(0, m_1, ID_1, upk_1, R_1^*)$, $k_1 = H_3(1, m_1, ID_1, upk_1, R_1^*)$, $Q = H_4(p, g, P_{pub})$ and $K_1 = (P_{pub})^{r_1} Q^{r_1}$. Then, $\mathcal{U}_1$ sends K_1 to $\mathcal{U}_2$.

(2) $\mathcal{U}_2$ randomly selects $r_2 \in Z_p^*$ and calculates $R_2^* = g^{r_2}$, $h_2 = H_2(0, m_2, ID_2, upk_2, R_2^*)$, $k_2 = H_3(1, m_2, ID_2, upk_2, R_2^*)$, $Q = H_4(p, g, P_{pub})$ and $K_2 = (P_{pub})^{r_2} Q^{r_2}$. Then, $\mathcal{U}_2$ sends K_2 to $\mathcal{U}_1$.

(3) $\mathcal{U}_1$ calculates $Y_1^* = (psk_1)^{h_1} \cdot K_2 \cdot Q^{k_1 x_1}$ and outputs an individual signature $\sigma_1^* = (R_1^*, Y_1^*)$ on m_1, where x_1 is the secret value of $\mathcal{U}_1$.

(4) $\mathcal{U}_2$ uses its secret value x_2 to calculate $Y_2^* = (psk_2)^{h_2} \cdot K_1 \cdot Q^{k_2 x_2}$ and outputs an individual signature $\sigma_2^* = (R_2^*, Y_2^*)$ on m_2.

(5) $\mathcal{U}_1$ cooperates with $\mathcal{U}_2$ to forge an aggregate signature $\sigma^* = (R_1^*, R_2^*, Y^*)$ on $\{m_1, m_2\}$, where $Y^* = Y_1^* \cdot Y_2^*$.

σ_i^* can easily be verified to be an invalid single signature for m_i under ID_i and upk_i since σ_i^* does not satisfy the individual signature verification equation, $i = 1, 2$. However, the

forged aggregate signature σ^* on $\{m_1, m_2\}$ under $\{ID_1, ID_2\}$ and $\{upk_1, upk_2\}$ is valid since

$$\begin{aligned} e(Y^*, g) &= e(Y_1^* Y_2^*, g) = e\left(\left(psk_1\right)^{h_1} K_2 Q^{k_1 x_1}\right. \\ &\cdot \left.\left(psk_2\right)^{h_2} K_1 Q^{k_2 x_2}, g\right) = e\left(\left(psk_1\right)^{h_1} K_1 Q^{k_1 x_1}\right. \\ &\cdot \left.\left(psk_2\right)^{h_2} K_2 Q^{k_2 x_2}, g\right) \\ &= e\left(\left(psk_1\right)^{h_1} \left(P_{pub}\right)^{r_1} Q^{r_1} Q^{k_1 x_1}\right. \\ &\cdot \left.\left(psk_2\right)^{h_2} \left(P_{pub}\right)^{r_2} Q^{r_2} Q^{k_2 x_2}, g\right) \\ &= e\left(R_1^* \left(Q_1\right)^{h_1}, P_{pub}\right) e\left(\left(upk_1\right)^{k_1} R_1^*, Q\right) \\ &\cdot e\left(R_2^* \left(Q_2\right)^{h_2}, P_{pub}\right) e\left(\left(upk_2\right)^{k_2} R_2^*, Q\right) \\ &= e\left(\prod_{i=1}^{2} R_i^* \left(Q_i\right)^{h_i}, P_{pub}\right) e\left(\prod_{i=1}^{2} \left(upk_i\right)^{k_i} R_i^*, Q\right). \end{aligned} \tag{3}$$

The above analysis shows that the forged aggregate signature σ^* satisfies the aggregate signature verification equation. Therefore, $\mathcal{U}_1$ can collude with $\mathcal{U}_2$ to produce a valid aggregate signature with some illegal single signatures. Therefore, Li et al.'s CLAS scheme [15] is vulnerable to coalition attacks.

4. Cryptanalysis of Du et al.'s CLAS Scheme

4.1. Review of Du et al.'s Scheme. In 2017, Du et al. [17] designed a CLAS scheme which is composed of the following seven algorithms:

(i) **Setup**: Given a security parameter $\lambda \in Z$, the KGC selects two cyclic groups G_1 and G_2 of prime order p, a generator g of G_1 and a bilinear pairing $e: G_1 \times G_1 \longrightarrow G_2$. Then, the KGC randomly selects $s \in Z_p^*$ and compute $P_{pub} = g^s$. Next, the KGC picks four hash functions $H_1, H_2, , H_3 : \{0,1\}^* \longrightarrow G_1$ and $H_4 : \{0,1\}^* \longrightarrow Z_p^*$. Finally, the KGC stores the master secret key $msk = s$ secret and publish public parameters $pp = \{G_1, G_2, p, g, , e, P_{pub}, H_1, H_2, H_3, H_4\}$.

(ii) **PartialKeyGen**: Given a user's identity ID_i, the KGC calculates $Q_i = H_1(ID_i)$ and $psk_i = (Q_i)^s$. Then, the KGC sends ID_i's partial private key psk_i to the user.

(iii) **UserKeyGen**: The user with identity ID_i randomly selects $x_i \in Z_p^*$ and calculates the public key $upk_i = g^{x_i}$. Then, the user sets the secret value $usk_i = x_i$ and the private key $sk_i = (psk_i, usk_i) = ((Q_i)^s, x_i)$.

(iv) **Sign**: Given a message m_i, a signer with identity ID_i executes the following steps:

(a) Randomly select $r_i \in Z_p^*$ and compute $R_i = g^{r_i}$.

(b) Compute $W_i = H_2(m_i, ID_i, upk_i)$, $T = H_3(P_{pub})$ and $h_i = H_4(m_i, ID_i, upk_i)$.

(c) Compute $Y_i = (psk_i)^{h_i} (W_i)^{x_i} T^{r_i}$.

(d) Output a signature $\sigma_i = (R_i, Y_i)$ on m_i.

(v) **Verify**: Given a signature $\sigma_i = (R_i, Y_i)$ on a message m_i under an identity ID_i and public key upk_i, a verifier first calculates $Q_i = H_1(ID_i)$, $W_i = H_2(m_i, ID_i, upk_i)$, $T = H_3(P_{pub})$ and $h_i = H_4(m_i, ID_i, upk_i)$. Then, the verifier verifies whether the following equation holds or not.

$$e(Y_i, g) = e\left(Q_i^{h_i}, P_{pub}\right) e(W_i, upk_i) e(T, R_i). \tag{4}$$

If it holds, then the verifier accepts σ_i; otherwise, the verifier rejects σ_i.

(vi) **Aggregate**: Given n message-signature pairs $\{(m_1, \sigma_1 = (R_1, Y_1)), \ldots, (m_n, \sigma_n = (R_n, Y_n))\}$ from n users, the aggregator calculates an aggregate signature $\sigma = (R, Y)$ on $\{m_1, \ldots, m_n\}$, where $R = \prod_{i=1}^{n} R_i$ and $Y = \prod_{i=1}^{n} Y_i$.

(vii) **AggregateVerify**: Given an aggregate signature $\sigma = (R, Y)$ on $\{m_1, \ldots, m_n\}$ under n identities $\{ID_1, \ldots, ID_n\}$ and n public keys $\{upk_1, \ldots, upk_n\}$, a verifier first calculates $Q_i = H_1(ID_i)$, $W_i = H_2(m_i, ID_i, upk_i)$, $T = H_3(P_{pub})$ and $h_i = H_4(m_i, ID_i, upk_i)$ for $1 \le i \le n$. Then, the verifier verifies whether the following equation holds or not.

$$e(Y, g) = e\left(\prod_{i=1}^{n} Q_i^{h_i}, P_{pub}\right) \prod_{i=1}^{n} e(W_i, upk_i) e(T, R). \tag{5}$$

If it holds, then the verifier accepts σ; otherwise, the verifier rejects σ.

4.2. Attack on Du et al.'s Scheme. Du et al. [17] claimed that their CLAS scheme is existentially unforgeable against the Type I and Type II adversaries in the random oracle model. However, we show that Du et al.'s CLAS scheme [17] is vulnerable to coalition attacks. Similarly, assuming $\mathcal{U}_1$ and $\mathcal{U}_2$ are two internal signers, they perform the following steps to generate two illegal single signatures, but the corresponding aggregate signature is legal.

(1) The signer $\mathcal{U}_1$ with identity ID_1 and public key upk_1 first selects a random value $r_1 \in Z_p^*$ and calculates $R_1^* = g^{r_1}$. Then, $\mathcal{U}_1$ randomly picks a message m_1, and calculates $W_1 = H_2(m_1, ID_1, upk_1)$, $T = H_3(P_{pub})$ and $h_1 = H_4(m_1, ID_1, upk_1)$. Finally, $\mathcal{U}_1$ sends T^{r_1} to $\mathcal{U}_2$.

(2) The signer $\mathcal{U}_2$ with identity ID_2 and public key upk_2 first selects a random value $r_2 \in Z_p^*$ and calculates $R_2^* = g^{r_2}$. Then, $\mathcal{U}_2$ randomly picks a message m_2, and calculates $W_2 = H_2(m_2, ID_2, upk_2)$, $T = H_3(P_{pub})$ and $h_2 = H_4(m_2, ID_2, upk_2)$. Finally, $\mathcal{U}_2$ sends T^{r_2} to $\mathcal{U}_1$.

(3) $\mathcal{U}_1$ calculates $Y_1^* = (psk_1)^{h_1} (W_1)^{x_1} T^{r_2}$ using the secret value x_1 and partial private key psk_1 and outputs an individual signature $\sigma_1^* = (R_1^*, Y_1^*)$ on m_1.

(4) $\mathcal{U}_2$ calculates $Y_2^* = (psk_2)^{h_2}(W_2)^{x_2}T^{r_1}$ using the secret value x_2 and partial private key psk_2 and outputs an individual signature $\sigma_2^* = (R_2^*, Y_2^*)$ on m_2.

(5) $\mathcal{U}_1$ cooperates with $\mathcal{U}_2$ to forge an aggregate signature $\sigma^* = (R^*, Y^*)$ on $\{m_1, m_2\}$, where $R^* = R_1^* \cdot R_2^*$ and $Y^* = Y_1^* \cdot Y_2^*$.

Obviously, σ_i^* is not a legal signature on message m_i under identity ID_i and public key upk_i, $i = 1, 2$. However, the forged aggregate signature σ^* on $\{m_1, m_2\}$ under $\{ID_1, ID_2\}$ and $\{upk_1, upk_2\}$ is valid since σ^* satisfies the following aggregate signature verification equation:

$$\begin{aligned} e(Y^*, g) &= e(Y_1^* Y_2^*, g) \\ &= e\left((psk_1)^{h_1}(W_1)^{x_1}T^{r_2} \cdot (psk_2)^{h_2}(W_2)^{x_2}T^{r_1}, g\right) \\ &= e\left((psk_1)^{h_1}(W_1)^{x_1}T^{r_1} \cdot (psk_2)^{h_2}(W_2)^{x_2}T^{r_2}, g\right) \\ &= e\left(Q_1^{h_1}, P_{pub}\right) e(W_1, upk_1)\, e(T, R_1^*) \\ &\quad \cdot e\left(Q_2^{h_2}, P_{pub}\right) e(W_2, upk_2)\, e(T, R_2^*) \\ &= e\left(\prod_{i=1}^{2} Q_i^{h_i}, P_{\text{pub}}\right) \prod_{i=1}^{2} e(W_i, upk_i)\, e(T, R^*). \end{aligned} \tag{6}$$

From the above analysis, we can see that $\mathcal{U}_1$ produces an illegal signature σ_1^* on message m_1, and $\mathcal{U}_2$ produces an illegal signature σ_2^* on message m_2, but the resulting aggregate signature σ^* on $\{m_1, m_2\}$ is valid. Hence, Du et al.'s CLAS scheme [17] is unable to withstand coalition attacks.

5. Cryptanalysis of Chen et al.'s CLAS Scheme

5.1. Review of Chen et al.'s Scheme. Chen et al.'s CLAS scheme [18] is described as follows:

(i) **Setup**: Given a security parameter $\lambda \in Z$, the KGC performs the following steps to generate the public parameters pp and the master secret key msk:

(a) Choose two cyclic groups G_1 and G_2 of prime order p, a generator g of G_1 and a bilinear pairing $e : G_1 \times G_1 \longrightarrow G_2$.

(b) Select a random integer $s \in Z_p^*$ and compute $P_{pub} = g^s$.

(c) Pick six cryptographic hash functions $H_1, H_1', H_2, H_3 : \{0,1\}^* \longrightarrow G_1$ and H_4, H_5: $\{0,1\}^* \longrightarrow Z_p^*$.

(d) Keep the master secret key $msk = s$ secret and publish public parameters $pp = \{G_1, G_2, p, g, e, P_{pub}, H_1, H_1', H_2, H_3, H_4, H_5\}$.

(ii) **PartialKeyGen**: Given a user's identity ID_i, the KGC computes $Q_{i,1} = H_1(ID_i)$ and $Q_{i,2} = H_1'(ID_i)$. Then, the KGC computes $psk_i = (psk_{i,1}, psk_{i,2}) = ((Q_{i,1})^s, (Q_{i,2})^s)$ and sends ID_i's partial private key psk_i to the user through a secure channel.

(iii) **UserKeyGen**: The user with identity ID_i generates a public/private key pair (upk_i, sk_i) according to the following steps:

(a) Select two random integers $x_{i,1}, x_{i,2} \in Z_p^*$ and compute the public key $upk_i = (upk_{i,1}, upk_{i,2}) = (g^{x_{i,1}}, g^{x_{i,2}})$.

(b) Set the secret value $usk_i = (x_{i,1}, x_{i,2})$ and the corresponding private key $sk_i = (psk_i, usk_i)$.

(iv) **Sign**: Given a message m_i and state information $st \in \{0,1\}^*$, a signer with identity ID_i performs the following:

(a) Select a random integer $r_i \in Z_p^*$ and compute $R_i = g^{r_i}$.

(b) Compute $W = H_2(st)$, $V = H_3(st)$, $h_i = H_4(m_i, ID_i, upk_i, pp)$, $h_i' = H_5(m_i, ID_i, upk_i, pp)$ and

$$Y_i = (psk_{i,1})^{h_i} \cdot psk_{i,2} \cdot W^{(h_i' x_{i,1} + x_{i,2})} \cdot V^{r_i}. \tag{7}$$

(c) Output a signature $\sigma_i = (R_i, Y_i)$ on m_i.

(v) **Verify**: Given an identity ID_i, a public key $upk_i = (upk_{i,1}, upk_{i,2})$, state information st, a message m_i, and a signature $\sigma_i = (R_i, Y_i)$, a verifier first computes $Q_{i,1} = H_1(ID_i)$, $Q_{i,2} = H_1'(ID_i)$, $W = H_2(st)$, $V = H_3(st)$, $h_i = H_4(m_i, ID_i, upk_i, pp)$ and $h_i' = H_5(m_i, ID_i, upk_i, pp)$. Then, the verifier checks the following equation:

$$\begin{aligned} e(Y_i, g) &= e\left((Q_{i,1})^{h_i} Q_{i,2}, P_{pub}\right) \\ &\quad \cdot e\left((upk_{i,1})^{h_i'} upk_{i,2}, W\right) \cdot e(R_i, V). \end{aligned} \tag{8}$$

If this equation holds, then the verifier accepts σ_i; otherwise, the verifier rejects σ_i.

(vi) **Aggregate**: Given n message-signature pairs $\{(m_1, \sigma_1 = (R_1, Y_1)), \ldots, (m_n, \sigma_n = (R_n, Y_n))\}$ from n users, the aggregator outputs an aggregate signature $\sigma = (R, Y)$ on messages $\{m_1, \ldots, m_n\}$, where $R = \prod_{i=1}^{n} R_i$ and $Y = \prod_{i=1}^{n} Y_i$.

(vii) **AggregateVerify**: Given n identities $\{ID_1, \ldots, ID_n\}$, n public keys $\{upk_1 = (upk_{1,1}, upk_{1,2}), \ldots, upk_n = (upk_{n,1}, upk_{n,2})\}$, state information st, and an aggregate signature $\sigma = (R, Y)$ on n messages $\{m_1, \ldots, m_n\}$, a verifier first computes $Q_{i,1} = H_1(ID_i)$, $Q_{i,2} = H_1'(ID_i)$, $W = H_2(st)$, $V = H_3(st)$, $h_i = H_4(m_i, ID_i, upk_i, pp)$, and $h_i' = H_5(m_i, ID_i, upk_i, pp)$ for $1 \le i \le n$. Then, the verifier checks the following equation:

$$\begin{aligned} e(Y, g) &= e\left(\prod_{i=1}^{n} (Q_{i,1})^{h_i} Q_{i,2}, P_{pub}\right) \\ &\quad \cdot e\left(\prod_{i=1}^{n} (upk_{i,1})^{h_i'} upk_{i,2}, W\right) \cdot e(R, V). \end{aligned} \tag{9}$$

If the equation holds, then the verifier accepts σ; otherwise, the verifier rejects σ.

5.2. Attack on Chen et al.'s CLAS Scheme. Chen et al. [18] demonstrated that their CLAS scheme was provably secure in the random oracle model. In this section, we present an attack on Chen et al.'s CLAS scheme [18] to demonstrate that their scheme cannot withstand an attack from a coalition of insider signers. Without loss of generality, we assume that $\{\mathcal{U}_1, \mathcal{U}_2\}$ are two signers with identities $\{ID_1, ID_2\}$ and corresponding public keys $\{upk_1 = (upk_{1,1}, upk_{1,2}), upk_2 = (upk_{2,1}, upk_{2,2})\}$, respectively. A message m_1 is signed by $\mathcal{U}_1$ with the private key $sk_1 = (psk_1, usk_1) = ((psk_{1,1}, psk_{1,2}), (x_{1,1}, x_{1,2}))$, and another message m_2 is signed by $\mathcal{U}_2$ with the private key $sk_2 = (psk_2, usk_2) = ((psk_{2,1}, psk_{2,2}), (x_{2,1}, x_{2,2}))$. In the following attack, $\mathcal{U}_1$ cooperates with $\mathcal{U}_2$ to generate invalid signatures $\{\sigma_1^*, \sigma_2^*\}$ on $\{m_1, m_2\}$, while the corresponding aggregate signature σ^* on $\{m_1, m_2\}$ satisfies the aggregate signature verification equation. Let st be the state information, $W = H_2(st)$ and $V = H_3(st)$.

(1) $\mathcal{U}_1$ chooses a random integer $r_1 \in Z_p^*$ and computes $R_1^* = g^{r_1}$, $h_1 = H_4(m_1, ID_1, upk_1, pp)$ and $h_1' = H_5(m_1, ID_1, upk_1, pp)$. Then, $\mathcal{U}_1$ sends V^{r_1} to $\mathcal{U}_2$.

(2) $\mathcal{U}_2$ chooses a random integer $r_2 \in Z_p^*$ and computes $R_2^* = g^{r_2}$, $h_2 = H_4(m_2, ID_2, upk_2, pp)$ and $h_2' = H_5(m_2, ID_2, upk_2, pp)$. Then, $\mathcal{U}_2$ sends V^{r_2} to $\mathcal{U}_1$.

(3) $\mathcal{U}_1$ computes $Y_1^* = (psk_{1,1})^{h_1} \cdot psk_{1,2} \cdot W^{h_1' x_{1,1} + x_{1,2}} \cdot V^{r_2}$ and sets $\sigma_1^* = (R_1^*, Y_1^*)$ as a signature on m_1.

(4) $\mathcal{U}_2$ computes $Y_2^* = (psk_{2,1})^{h_2} \cdot psk_{2,2} \cdot W^{h_2' x_{2,1} + x_{2,2}} \cdot V^{r_1}$ and sets $\sigma_2^* = (R_2^*, Y_2^*)$ as a signature on m_2.

(5) $\mathcal{U}_1$ cooperates with $\mathcal{U}_2$ to generate a forged aggregate signature $\sigma^* = (R^*, Y^*)$ on $\{m_1, m_2\}$, where $R^* = R_1^* R_2^*$ and $Y^* = Y_1^* Y_2^*$.

Clearly, $\sigma_1^* = (R_1^*, Y_1^*)$ is an invalid single signature on m_1 since it does not satisfy the verification equation as follows:

$$\begin{aligned} e(Y_1^*, g) &= e\left((psk_{1,1})^{h_1}\, psk_{1,2} W^{h_1' x_{1,1} + x_{1,2}} V^{r_2}, g\right) \\ &= e\left((psk_{1,1})^{h_1}\, psk_{1,2}, g\right) e\left(W^{h_1' x_{1,1} + x_{1,2}}, g\right) e\left(g^{r_2}, V\right) \\ &= e\left(H_1(ID_1)^{h_1} H_1'(ID_1), P_{pub}\right) \\ &\quad \cdot e\left((upk_{1,1})^{h_1'} upk_{1,2}, W\right) \cdot e\left(R_2^*, V\right) \\ &\neq e\left((Q_{1,1})^{h_1} Q_{1,2}, P_{pub}\right) \cdot e\left((upk_{1,1})^{h_1'} upk_{1,2}, W\right) \\ &\quad \cdot e\left(R_1^*, V\right) \end{aligned} \tag{10}$$

Similarly, the single signature $\sigma_2^* = (R_2^*, Y_2^*)$ on m_2 is also invalid. However, $\sigma^* = (R^*, Y^*)$ is a valid aggregate signature on $\{m_1, m_2\}$. The correctness of σ^* can be verified as follows:

$$\begin{aligned} e(Y^*, g) &= e(Y_1^* Y_2^*, g) \\ &= e\left((psk_{1,1})^{h_1}\, psk_{1,2} W^{h_1' x_{1,1} + x_{1,2}} V^{r_2} \right. \\ &\quad \left. \cdot (psk_{2,1})^{h_2}\, psk_{2,2} W^{h_2' x_{2,1} + x_{2,2}} V^{r_1}, g\right) \\ &= e\left((psk_{1,1})^{h_1}\, psk_{1,2} W^{h_1' x_{1,1} + x_{1,2}} V^{r_1} \right. \\ &\quad \left. \cdot (psk_{2,1})^{h_2}\, psk_{2,2} W^{h_2' x_{2,1} + x_{2,2}} V^{r_2}, g\right) \\ &= e\left(\prod_{i=1}^{2} (Q_{i,1})^{h_i} Q_{i,2}, P_{pub}\right) \\ &\quad \cdot e\left(\prod_{i=1}^{2} (upk_{i,1})^{h_i'} upk_{i,2}, W\right) \cdot e\left(R^*, V\right). \end{aligned} \tag{11}$$

Hence, in Chen et al.'s CLAS scheme [18], dishonest insider signers can cooperate to forge valid aggregate signatures of arbitrary messages by combining invalid individual signatures. This result also shows that the validity of an aggregate signature does not guarantee the validity of every signature involved in the aggregation. Therefore, our attack is successful; that is, Chen et al.'s CLAS scheme [18] is insecure against coalition attacks.

6. Improved Certificateless Aggregate Signature Scheme

In Chen et al.'s CLAS scheme [18], Du et al.'s CLAS scheme [17], and Li et al.'s CLAS scheme [15], the main reason that the abovementioned coalition attacks can occur is that insider signers can exchange some information involved in the individual signature generation, such as h^{r_i} and $(P_{pub})^{r_i} Q^{r_i}$. To overcome these security weaknesses, we propose an improved CLAS scheme based on Chen et al.'s CLAS scheme [18]. Our CLAS scheme is described as follows:

(i) The **Setup**, **PartialKeyGen**, and **UserKeyGen** algorithms are the same as those of Chen et al.'s CLAS scheme described in Section 4. The only difference is that the **Setup** algorithm adds a collision-resistant hash function $H_6 : \{0,1\}^* \longrightarrow \{0,1\}^l$, where l is the length of the output message of H_6.

(ii) **Sign**: Given a message m_i and state information $st \in \{0,1\}^*$, a signer with identity ID_i executes the following steps:

(a) Randomly select $r_i \in Z_p^*$ and compute $R_i = g^{r_i}$.

(b) Compute $W = H_2(st)$, $V = H_3(st)$, $h_i = H_4(m_i, ID_i, upk_i, R_i, W, V, pp)$, $h_i' = H_5(m_i, ID_i, upk_i, R_i, W, V, pp)$ and

$$Y_i = (psk_{i,1})^{h_i} \cdot psk_{i,2} \cdot W^{(h_i' x_{i,1} + x_{i,2} + h_i r_i)} \cdot V^{r_i}. \tag{12}$$

(c) Output $\sigma_i = (R_i, Y_i)$ as a signature on m_i.

(iii) **Verify**: This algorithm is the same as the **Verify** algorithm in Chen et al.'s CLAS scheme, except that the signature verification equation is modified as follows:

$$e(Y_i, g) = e\left((Q_{i,1})^{h_i} Q_{i,2}, P_{pub}\right) \cdot e\left((upk_{i.1})^{h_i'} upk_{i,2} (R_i)^{h_i}, W\right) \cdot e(R_i, V), \tag{13}$$

where $h_i = H_4(m_i, ID_i, upk_i, R_i, W, V, pp)$ and $h_i' = H_5(m_i, ID_i, upk_i, R_i, W, V, pp)$.

(iv) **Aggregate**: Given n message-signature pairs $\{(m_1, \sigma_1 = (R_1, Y_1)), \ldots, (m_n, \sigma_n = (R_n, Y_n))\}$ from n signers, the aggregator first computes

$$Y = H_6(e(Y_1, g), \ldots, e(Y_n, g)) \tag{14}$$

and then outputs an aggregate signature $\sigma = (R_1, \ldots, R_n, Y)$ on messages $\{m_1, \ldots, m_n\}$.

(v) **AggregateVerify**: Given n identities $\{ID_1, \ldots, ID_n\}$, n public keys $\{upk_1 = (upk_{1,1}, upk_{1,2}), \ldots, upk_n = (upk_{n,1}, upk_{n,2})\}$, state information st and an aggregate signature $\sigma = (R_1, \ldots, R_n, Y)$ on n messages $\{m_1, \ldots, m_n\}$, a verifier first computes $Q_{i,1} = H_1(ID_i)$, $Q_{i,2} = H_1'(ID_i)$, $W = H_2(st)$, $V = H_3(st)$, $h_i = H_4(m_i, ID_i, upk_i, R_i, W, V, pp)$ and $h_i' = H_5(m_i, ID_i, upk_i, R_i, W, V, pp)$ for $1 \le i \le n$. Then, the verifier checks whether

$$\begin{aligned} Y = {} & H_6\left(e\left((Q_{1,1})^{h_1} Q_{1,2}, P_{pub}\right)\right. \\ & \cdot e\left((upk_{1,1})^{h_1'} upk_{1,2} (R_1)^{h_1}, W\right) \\ & \cdot e(R_1, V), \ldots, e\left((Q_{n,1})^{h_n} Q_{n,2}, P_{pub}\right) \\ & \left. \cdot e\left((upk_{n,1})^{h_n'} upk_{n,2} (R_n)^{h_n}, W\right) \cdot e(R_n, V)\right) \end{aligned} \tag{15}$$

holds. If it holds, then the verifier accepts σ; otherwise, the verifier rejects σ.

If H_i is a collision-resistant hash function, then it is difficult to find z_1 and z_2 such that $H_i(z_1) = H_i(z_2)$ [24]. In our improved CLAS scheme, if the KGC selects $t_1, t_2 \in Z_p^*$ and computes $W = g^{t_1}$ and $V = g^{t_2}$, then the KGC is unable to find a unique st to satisfy $H_2(st) = W$ and $H_3(st) = V$ due to the collision resistance of H_2 and H_3. However, if the KGC computes $W = H_2(st) = g^{t_1}$ and $V = H_3(st) = g^{t_2}$, then the difficulty of the KGC calculating $t_1, t_2 \in Z_p^*$ from W and V is equivalent to solving the discrete logarithm problem. Therefore, the improved CLAS scheme can resist malicious-but-passive KGC attacks unless the adversary can break the collision resistance of H_2 and H_3.

The part R_i in every individual signature is embedded in hash values $h_i = H_4(m_i, ID_i, upk_i, R_i, W, V, pp)$ and $h_i' = H_5(m_i, ID_i, upk_i, R_i, W, V, pp)$, which makes it impossible for an attacker to modify R_i at will. Hence, our improved CLAS scheme is resistant to the universal forgery attacks in [19–21].

The unforgeability of the proposed CLAS scheme is almost the same as that of Chen et al.'s CLAS scheme. Since Chen et al. [18] demonstrated that their CLAS scheme is existentially unforgeable in the random oracle model, our improved CLAS scheme can be similarly proven to be existentially unforgeable in the random oracle model. The security proofs of the two schemes are basically the same, so we omit the detailed description. Here, we prove only that the proposed CLAS scheme can resist the coalition attack described in Section 4.

Theorem 1. *If the hash function H_6 is collision-resistant, then the proposed CLAS scheme can withstand attacks from the coalition of insider signers.*

Proof. If each signature $\sigma_i = (R_i, Y_i)$ involved in the aggregation is valid, we have that

$$e(Y_i, g) = e\left((Q_{i,1})^{h_i} Q_{i,2}, P_{pub}\right) \cdot e\left((upk_{i,1})^{h_i'} upk_{i,2} (R_i)^{h_i}, W\right) \cdot e(R_i, V) \quad \text{for } i = 1, \ldots, n. \tag{16}$$

Thus, we can obtain

$$\begin{aligned} Y &= H_6(e(Y_1, g), \ldots, e(Y_n, g)) \\ &= H_6\left(e\left((Q_{1,1})^{h_1} Q_{1,2}, P_{pub}\right)\right. \\ &\quad \cdot e\left((upk_{1,1})^{h_1'} upk_{1,2} (R_1)^{h_1}, W\right) \\ &\quad \cdot e(R_1, V), \ldots, e\left((Q_{n,1})^{h_n} Q_{n,2}, P_{pub}\right) \\ &\quad \left. \cdot e\left((upk_{n,1})^{h_n'} upk_{n,2} (R_n)^{h_n}, W\right) \cdot e(R_n, V)\right). \end{aligned} \tag{17}$$

Therefore, the corresponding aggregate signature $\sigma = (R_1, \ldots, R_n, Y)$ is also valid.

Meanwhile, if an aggregate signature $\sigma = (R_1, \ldots, R_n, Y)$ is valid, we have

$$\begin{aligned} Y = {} & H_6\left(e\left((Q_{1,1})^{h_1} Q_{1,2}, P_{pub}\right)\right. \\ & \cdot e\left((upk_{1,1})^{h_1'} upk_{1,2} (R_1)^{h_1}, W\right) \\ & \cdot e(R_1, V), \ldots, e\left((Q_{n,1})^{h_n} Q_{n,2}, P_{pub}\right) \\ & \left. \cdot e\left((upk_{n,1})^{h_n'} upk_{n,2} (R_n)^{h_n}, W\right) \cdot e(R_n, V)\right) \\ = {} & H_6(e(Y_1, g), \ldots, e(Y_n, g)). \end{aligned} \tag{18}$$

Since the hash function H_6 is collision-resistant, $H_6(V^{r_i}, V^{r_j})$ is not equal to $H_6(V^{r_j}, V^{r_i})$ for any two values V^{r_i} and V^{r_j}.

From the above aggregate signature verification equation, we derive the following equations:

$$e(Y_i, g) = e\left((Q_{i,1})^{h_i} Q_{i,2}, P_{pub}\right) \cdot e\left((upk_{i,1})^{h_i'} upk_{i,2} (R_i)^{h_i}, W\right) \cdot e(R_i, V) \quad \text{for } i = 1, \ldots, n. \tag{19}$$

This derivation implies that an aggregate signature $\sigma = (R_1, \ldots, R_n, Y)$ is valid if and only if each signature $\sigma_i = (R_i, Y_i)$ involved in the aggregation is valid. Therefore, the proposed CLAS scheme can withstand forgery attacks from a coalition of insider signers. □

7. Conclusion

In this paper, we present a cryptanalysis of three CLAS schemes. The analysis shows that Li et al.'s CLAS scheme [15], Du et al.'s CLAS scheme [17], and Chen et al.'s CLAS scheme [18] are insecure against attacks from a coalition of insider signers. To overcome these attacks, we propose an improved CLAS scheme. Compared with Chen et al.'s CLAS scheme [18], our improved scheme has large computational overhead in generating and verifying aggregate signatures. How to construct a secure CLAS scheme without bilinear pairing is a challenging issue, which we leave as an open problem.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (61662069), the China Postdoctoral Science Foundation (2017M610817), the Natural Science Foundation of Gansu Province of China (1506RJZA130), the Science and Technology Project of Lanzhou City of China (2013-4-22), and the Foundation for Excellent Young Teachers by Northwest Normal University (NWNU-LKQN-14-7).

References

[1] L. Cheng, Q. Wen, Z. Jin, H. Zhang, and L. Zhou, "Cryptanalysis and improvement of a certificateless aggregate signature scheme," *Information Sciences*, vol. 295, pp. 337–346, 2015.

[2] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology: Proceedings of (CRYPTO '84)*, vol. 196 of *Lecture Notes in Computer Science*, pp. 47–53, Springer, Berlin, Germany, 1985.

[3] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Advances in Cryptology-ASIACRYPT*, vol. 2894 of *Lecture Notes in Computer Science*, pp. 452–473, Springer, 2003.

[4] X. Huang, Y. Mu, W. Susilo et al., "Certificateless signature revisited," *Information Security and Privacy*, pp. 308–322, November 2007.

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

[6] S. K. H. Islam and G. P. Biswas, "Provably secure and pairing-free certificateless digital signature scheme using elliptic curve cryptography," *International Journal of Computer Mathematics*, vol. 90, no. 11, pp. 2244–2258, 2013.

[7] L. Wang, K. Chen, Y. Long, X. Mao, and H. Wang, "A Modified Efficient Certificateless Signature Scheme without Bilinear Pairings," in *Proceedings of the 2015 International Conference on Intelligent Networking and Collaborative Systems (INCOS)*, pp. 82–85, Taipei, Taiwan, September 2015.

[8] K. Yeh, "Cryptanalysis of Wang et al.'s certificateless signature scheme without bilinear pairings," *in IACR Cryptology ePrint Archive*, p. 217, 2017.

[9] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Advances in Cryptology (EUROCRYPT)*, pp. 416–432, Springer, Berlin, Germany, 2003.

[10] F. Zhang, L. Shen, and G. Wu, "Notes on the security of certificateless aggregate signature schemes," *Information Sciences*, vol. 287, pp. 32–37, 2014.

[11] H. Liu, S. Wang, M. Liang, and Y. Chen, "New construction of efficient certificateless aggregate signatures," *International Journal of Security and Its Applications*, vol. 8, no. 1, pp. 411–422, 2014.

[12] H. Liu, M. Liang, and H. Sun, "A secure and efficient certificateless aggregate signature scheme," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E97-A, no. 4, pp. 991–995, 2014.

[13] H. Xiong, Z. Guan, Z. Chen, and F. Li, "An efficient certificateless aggregate signature with constant pairing computations," *Information Sciences*, vol. 219, pp. 225–235, 2013.

[14] D. He, M. Tian, and J. Chen, "Insecurity of an efficient certificateless aggregate signature with constant pairing computations," *Information Sciences*, vol. 268, pp. 458–462, 2014.

[15] J. Li, H. Yuan, and Y. Zhang, "Cryptanalysis and improvement for certificateless aggregate signature," *Fundamenta Informaticae*, vol. 157, no. 1-2, pp. 111–123, 2018.

[16] Y. Ming, X. Zhao, and Y. Wang, "Certificateless aggregate signature scheme," *Journal of University of Electronic Science and Technology of China*, vol. 43, no. 2, pp. 188–193, 2014.

[17] H. Z. Du and Q. Y. Wen, "Attack and improvement of a certificateless aggregate signature scheme," *Acta Scientiarum Naturalium Universitatis Sunyatseni. Zhongshan Daxue Xuebao. Ziran Kexue Ban*, vol. 56, no. 1, pp. 77–84, 2017.

[18] Y.-C. Chen, R. Tso, M. Mambo, K. Huang, and G. Horng, "Certificateless aggregate signature with efficient verification," *Security and Communication Networks*, vol. 8, no. 13, pp. 2232–2243, 2015.

[19] H. Chen, S. M. Wei, C. J. Zhu, and Y. Yang, "Secure certificateless aggregate signature scheme," *Journal of Software. Ruanjian Xuebao*, vol. 26, no. 5, pp. 1173–1180, 2015.

[20] H. Zhang, "Insecurity of a certificateless aggregate signature scheme," *Security and Communication Networks*, vol. 9, no. 11, pp. 1547–1552, 2016.

[21] H. Shen, J. Chen, J. Shen et al., "Cryptanalysis of a certificateless aggregate signature scheme with efficient verification," *Security and Communication Networks*, vol. 9, no. 13, pp. 2217–2221, 2016.

[22] L. Shen, J. Ma, X. Liu, F. Wei, and M. Miao, "A Secure and Efficient ID-Based Aggregate Signature Scheme for Wireless Sensor Networks," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 546–554, 2017.

[23] M. H. Au, J. Chen, J. K. Liu, Y. Mu, D. S. Wong, and G. Yang, "Malicious KGC attacks in certificateless cryptography," in *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security, ASIACCS '07*, pp. 302–311, March 2007.

[24] P. Rogaway and T. Shrimpton, "Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 3017, pp. 371–388, 2004.