Security Vulnerabilities of Four Signature Schemes From NTRU Lattices and Pairings

KYUNG-AH SHIM, (Member, IEEE)
National Institute for Mathematical Sciences, Daejeon 34047, South Korea
e-mail: kashim@nims.re.kr
This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) under Grant NRF-2017R1E1A1A03070068.

ABSTRACT Certificateless cryptography solves the certificate management problem in public-key cryptography and the key-escrow problem in identity-based cryptography. Xie et al. proposed a certificateless signature scheme using NTRU lattices. They proved that their scheme was existential unforgeable for Type I and Type II adversaries under the intractability of the Small Integer Solution (SIS) problem on NTRU lattices in the random oracle model. Subsequently, Hung et al. proposed a revocable certificateless signature scheme on NTRU lattices that gave a revocation method via public channels to revoke illegal or compromised users. They also proved that their scheme was existential unforgeable for Type I, Type II and Type III adversaries under the intractability of the SIS problem. Recently, Wang et al. proposed a certificateless aggregate signature scheme from bilinear pairings and Rezaeibagha et al. proposed a new lightweight certificateless scheme for Industrial Internet of Things from bilinear pairings. The security of their schemes was proven for Type I and Type II adversaries under the intractability of some mathematical problems. In this paper, we show that the two schemes on NTRU lattices are insecure against Type I or Type II adversaries. We also point out invalidity of Wang et al.’s scheme and vulnerability of Rezaeibagha et al.’s scheme against Type I attacks. We then suggest some improvements to prevent our attacks.

INDEX TERMS Certificateless signature scheme, certificateless aggregate signature scheme, revocation, type I adversary, type II adversary.

I. INTRODUCTION Public-key cryptography using a random string as a public key needs authentication of the public key. The authentication of the public key is realized by a public key certificate issued from a Certificate Authority in Public-Key Infrastructure (PKI). PKI suffers from the certificate management problem including distribution, computational overhead for certificate verification and revocation. To solve the problem, Shamir [12] proposed the concept of identity (ID)-based cryptography, where users’ public keys can be calculated from unique identity information. Since Key Generation Center (KGC) or Private Key Generator (PKG) with a master secret key generates users’ private keys in ID-based schemes, the schemes cause the key-escrow problem. To solve the certificate management problem in public-key cryptography and the key-escrow problem identity-based cryptography, Al-Riyami and Paterson [1] proposed a new concept of certificateless cryptography. Certificateless cryptography solves the problems by generating users’ secret keys as the combination of some user-chosen secret values and secret information of KGC with a master secret key.

Signature schemes, GGH [4] and NTRUSign [5], using NTRU lattices were proposed, but were completely broken in [10] due to the leakage of some secret information for their lattice trapdoors. To prevent such a leakage, Gentry et al. [3] proposed a new signature scheme (GVP scheme) secure under the hardness assumptions of worst-case lattice problems. At Eurocrypt 2012, Lyubashevsky and Micciancio [8] gave an efficient signature scheme based on the Short Integer Solution (SIS) problem and the LWE problem with security reductions to the worst-case assumptions in general lattices. Lyubashevsky’s scheme was constructed from the Fiat-Shamir transformation and its signing algorithm required sampling from discrete Gaussian. Lyubashevsky [9] proposed a more efficient signature scheme.
using the rejection sampling technique for signing. There have been proposed ID-based signature schemes [13], [15] and a certificateless signature scheme [14] using the GPV scheme [3] to extract users’ partial private keys. To improve efficiency, Xie et al. [18] proposed a new CLS scheme on NTRU lattices and proved its existential unforgeability against Type I and Type II adversaries under the intractability of the SIS problem on NTRU lattices in the random oracle model. Huang et al. [7] proposed a revocable CLS (RCLS) scheme on NTRU lattices that gave a revocation method via public channels to revoke illegal or compromised users. Their scheme utilized a sampling algorithm of the key extraction technique in the ring variant of Lyubashevsky’s signature scheme [9] to generate signatures. They showed that their scheme was existential unforgeable for Type I, Type II and Type III adversaries under the intractability of the SIS problem. Recently, Wang et al. [17] proposed a certificateless aggregate signature (CLAS) scheme from bilinear pairings and Rezaeibagha et al. [11] proposed a new lightweight CLS scheme for Industrial Internet of Things from bilinear pairings. The security of their schemes was proven for Type I and Type II adversaries under the intractability of some mathematical problems. In this paper, we show that Huang et al.’s revocable CLS scheme and Xie et al.’s CLS scheme and are insecure against Type I or Type II adversaries despite their security proofs. We also point out invalidity of Wang et al.’s CLAS scheme and vulnerability of Rezaeibagha et al.’s CLS scheme against Type I attacks. We then suggest some improvements to prevent our attacks.

Outline of this paper is as follows. We introduce Hung et al.’s revocable CLS scheme and Xie et al.’s CLS scheme on NTRU lattices and present Type I or Type II attacks in Section II. We describe Wang et al.’s scheme and Rezaeibagha et al.’s scheme and point out invalidity of Wang et al.’s scheme and vulnerability of Rezaeibagha et al.’s scheme against Type I attacks in Section III. In Section VI, we suggest some improvements to prevent our attacks. We conclude this paper in Section V.

II. VULNERABILITIES OF TWO CLS SCHEMES FROM LATTICES
In this section, we describe Hung et al.’s revocable CLS scheme [7] and Xie et al.’s CLS scheme [18] and present Type I or Type II attacks on the schemes.

1) NOTATIONS
- \( Z \): the set of integers.
- \( Z_q \): the set of integers in \([-q/2, q/2]\) for \( q > 0 \).
- \( R_q = Z_q[x]/(x^N + 1) \): a polynomial ring of modulo \( x^N + 1 \) defined on \( Z_q \).
- \( ||x|| = \sqrt{\sum x_i^2} \) for a vector \( x \).
- \( ||v||_1 \): the number of nonzero elements of a vector \( v \) for \( v : v \in \{-1, 0, 1\}^N \).
- \( f = \sum_{i=1}^{N-1} f_i x^i, \ g = \sum_{i=1}^{N-1} g_i x^i \): two polynomials in \( R_q \).

A. DESCRIPTION OF TWO CLS SCHEMES ON NTRU LATTICES
Hung et al.’s revocable CLS scheme and Xie et al.’s CLS scheme are based on the combination of the key extraction in Ducas et al.’s ID-based encryption [2] and a ring variant of Lyubashevsky’s signature scheme [9].

1) HUNG et al.’S REVOCABLE CLS SCHEME
Hung et al. [7] constructed a revocable CLS (RCLS) scheme on NTRU lattices. In Hung et al.’s RCLS scheme, a user’s private key consists of a partial private key, a secret value and a time update key. The user selects the secret value and KGC generates the partial private key from the user’s identity. KGC publishes the public parameters as \( KGC \) securely.

KGC selects \( a_1, a_2 \in Z_q \) as a system public key and sets \( S_{KGC} = B \) as a system secret key.

KGC selects three collision-resistant hash functions \( H_0, H_1 : \{0, 1\}^* \rightarrow Z_q^N \) and \( H_2 : Z_q^N \times Z_q^N \times \{0, 1\}^* \rightarrow \{v : v \in \{-1, 0, 1\}^N, ||v||_1 \leq \lambda \} \).

KGC publishes the public parameters as

\[ \text{Parms} = (N, s, \sigma, \lambda, q, h, a_1, a_2, H_0, H_1, H_2) \]

- Partial private key extract: For a user identity \( ID \in \{0, 1\}^* \),

KGC generates a partial private key \( D_{ID} = (s_1, s_2) \) such that

\[ s_1 + h * s_2 = P_{ID} = H_0(ID) \in Z_q^N \]

and \( ||(s_1, s_2)|| < s \sqrt{2N} \) by performing \( \text{SampleGau}(B, s, (P_{ID}, 0)) \) in [2], where \( P_{ID} \) is the first partial public key.

Then KGC sends \( D_{ID} = (s_1, s_2) \) to the user securely.

- Time key update: For a time period \( t \) and a user identity \( ID \),

KGC calculates a time update key such that

\[ s_3 + h * s_4 = T_{ID}, T_{ID} = H_1(ID, t) \in Z_q^N \]
and \( \|s_2, s_4\| < s\sqrt{2N} \) by performing \texttt{SampleGau}(B, s, (T_{ID,}, 0)) in [2].
- Then, KGC transmits the time update key \( T_{ID,t} = (s_3, s_4) \) to the user through a secure channel.

### Set secret value:
- For an identity \( ID \),
- A user with ID selects random values \( s_5, s_6 \) uniformly from \([-d, \ldots, d]\) and sets \( S_{ID} = (s_5, s_6) \) as a secret value, where \( 1 \leq d \leq 31 \).
- The user computes

\[
R_{ID} = a_1 * s_5 + a_2 * s_6
\]

as a second partial public key.

### Set private key:
- A user with ID sets \( S_{KID} = (D_{ID}, T_{ID,t}, S_{ID}) \) as the private key.

### Set public key:
- A user with ID sets \( PK_{ID} = (P_{ID}, R_{ID}) \) as the public key.

### Sign:
- Given a message \( \mu \in \{0, 1\}^* \),
- A signer with the private key \( S_{KID} \) selects random values \( y_1, y_2, y_3, y_4, y_5, y_6 \) by the distribution \( D_{\sigma}^N \), and computes

\[
c = H_2(y_1 + h * y_2, y_3 + h * y_4, a_1 * y_5 + a_2 * y_6, \mu),
\]
- \( z_1 = y_1 + s_1 * c, z_2 = y_2 + s_2 * c, z_3 = y_3 + s_3 * c, z_4 = y_4 + s_4 * c, z_5 = y_5 + s_5 * c, z_6 = y_6 + s_6 * c, \)
- where \( \|z_1, z_2, z_3, z_4, z_5, z_6\| \leq 2\sigma\sqrt{6N} \).
- If no such \( (z_1, z_2, z_3, z_4, z_5, z_6) \) is generated, repeat this algorithm. This procedure is the rejection sampling technique. At last, a constant \( M = O(1) \) is determined so that a signature \( (z_1, z_2, z_3, z_4, z_5, z_6) \) can be produced with probability

\[
\min(D_{\sigma}^N(z), 1),
\]
- in the ring variant of Lyubashevsky’s signature scheme [9], where

\[
z = [y_1^T || y_2^T || y_3^T || y_4^T || y_5^T || y_6^T]^T
\]
and

\[
v = [(s_1 * c)^T || (s_2 * c)^T || (s_3 * c)^T || (s_4 * c)^T || (s_5 * c)^T || (s_6 * c)^T]^T.
\]

### Verify:
- For a signature \( (z = (z_1, z_2, z_3, z_4, z_5, z_6), c) \) on a message \( \mu \) associated to ID, a verifier checks the following equality

\[
c = H_2(z_1 + h * z_2 - P_{ID} * c, z_3 + h * z_4 - T_{ID} * c, a_1 * z_5 + a_2 * z_6 - R_{ID} * c, \mu).
\]
- If it holds, output valid.

2) \text{XIE et al.’s CLS SCHEME}

Xie \textit{et al.} [18] constructed a provably secure CLS scheme on NTRU lattices. Their CLS scheme consisting of seven polynomial-time algorithms runs as follows:

- **Setup:** KGC generates a master secret key and a master public key as in Hung \textit{et al.}’s RCLS scheme.
  - KGC sets \( msk = B \) as the master secret key and \( mpk = h \) as the master public key.

- **Partial private key extract:** For a user identity \( id \in \{0, 1\}^* \),
  - KGC with the master secret key \( msk \) generates a partial private key \((s_1, s_2)\) by running \texttt{Sample-Gau}(B, s, (H(id), 0)) in [2]. Then KGC sends \((s_1, s_2)\) to the user via a secure way.
  - After receiving \((s_1, s_2)\), the user checks the following equations

\[
||s_1, s_2|| \leq s\sqrt{2N} \text{ and } s_1 + s_2 * h = H(id).
\]
- If they hold, the user accepts \((s_1, s_2)\) as the partial private key \( d_{id} \).

- **Set secret value:** For a user with \( id \),
  - A user with \( id \) picks random values \( s_1, s_2 \in \mathbb{Z}_{2p}, s \) and returns \( s_{id} = (s_1', s_2') \) as a secret value.

- **Set private key:** For a user with \( id \),
  - A user with \( id \) is sets \( sk_{id} = (d_{id}, s_{id}) \) as a full private key.

- **Set public key:** For a user with \( id \),
  - A user with \( id \) sets \( pk_{id} = s_1' + s_2' * h \) as a public key.

- **CL-Sign:** For a message \( \mu \in \{0, 1\}^* \),
  - A signer with the private key \( sk_{id} \) chooses random \( y_1, y_2, y_1', y_2' \in \mathbb{Z}_{2p}, s \) and computes

\[
e = H_1(y_1 + y_2 * h, y_1' + y_2' * h, \mu), \]
- \( z_1 = y_1 + s_1 * e, z_2 = y_2 + s_2 * e, \)
- \( z_1' = y_1' + s_1' * e, z_2' = y_2' + s_2' * e. \)
- As in [9], \((z_1, z_2, z_1', z_2')\) is generated with probability

\[
\min(D_{\mathbb{Z}_{2p}^*, \sigma}(z), 1).
\]
- In the form variant of Lyubashevsky’s signature scheme [9], where

\[
z = [y_1^T || y_2^T || y_1'^T || y_2'^T]^T
\]

and

\[
v = [(s_1 * e)^T || (s_2 * e)^T || (s_1' * e)^T || (s_2' * e)^T]^T.
\]

### B. VULNERABILITIES OF TWO CLS SCHEMES FROM LATTICES

Now, we show Huang \textit{et al.}’s RCLS scheme [7] and Xie \textit{et al.}’s CLS scheme [18] are insecure against Type I or Type II adversaries despite their security proofs for existential unforgeability under the hardness of the SIS problem.

1) **ATTACKS ON HUNG \textit{et al.}’S RCLS SCHEME**

For security analysis of Hung \textit{et al.}’s RCLS scheme, we review the security and adversarial models of the RCLS...
schemes. Adversaries in RCLS schemes consist of the following three Types [6], [16]:

- **Type I adversary (outsider):** An adversary knows the secret value and the time update key of any user obtained by replacing the associated public key and listening to the public channel.
- **Type II adversary (honest-but-curious KGC):** An adversary knows the time update key and the partial private key of any user, but it cannot know the secret value associated to the user.
- **Type III adversary (revoked user):** An adversary knows the secret value and the partial private key of any user, but it cannot know the current time update key associated to the user.

An RCLS scheme is existential unforgeable against adaptive chosen-message attacks if a polynomial-time adversary wins the following RCLS-UF-ACMA game between $A$ and a challenger $C$ with a non-negligible advantage:

- **Setup.** The challenger $C$ generates the public parameters $\text{Parms}$ and the system secret key $S_{\text{KGC}}$ which is kept secret for $C$.
  - KGC is the challenger $C$ for Type I and III adversaries.
  - The honest-but-curious KGC is a Type II adversary. Thus, $A$ knows the system secret key $S_{\text{KGC}}$ and can calculate the time update key and the partial private key of any user.
- **Queries.** $A$ makes the following queries to $C$ adaptively:
  - **Partial private key extract queries (ID).** For a query associated to a user identity $ID$, $C$ returns the partial private key $D_{ID}$ to $A$ after running the partial private key extract algorithm.
  - **Time key update queries (ID, $t$).** For a query associated to a time period $t$ and a user identity $ID$, $C$ returns the time update key $T_{ID,t}$ to $A$ after running the time key update algorithm.
  - **Secret value queries (ID).** For a query associated to a user identity $ID$, $C$ returns the user secret value $S_{ID}$ to $A$ after running the set secret value algorithm.
  - **Public key queries (ID).** For a query associated to a user identity $ID$, $C$ returns the user public key $PK_{ID}$ to $A$.
  - **Public key replacement queries (ID, $PK_{ID}$).** For a query associated to a user identity $ID$ and a new public key $PK_{ID}'$, $C$ keeps such replacement in the list.
  - **Sign queries (ID, $PK_{ID}$, $m$, $t$).** For a query on a message $m$ associated to a user identity $ID$, and a time period $t$, $C$ runs the sign algorithm to generate a signature $z$ on $m$ and outputs $z$ to $A$.
- **Forgery.** At last, the challenger $C$ outputs $$(ID^*, PK_{ID}, m^*, z^*, t^*),$$ where $ID^*$ is a target identity. $C$ wins the RCLS-UF-ACMA game if the conditions are satisfied:
  - $(ID^*, m^*, t^*)$ was never requested to the signing queries.
  - $(ID^*, PK_{ID}, m^*, z^*, t^*)$ in the Verify algorithm is valid.
  - For a Type I adversary $C$, $ID^*$ was never requested to the partial private key extract queries.
  - For a Type II adversary $C$, $ID^*$ was never requested to the public key replacement queries and the secret value queries.
  - For a Type III adversary $C$, $(ID^*, t^*)$ was never requested to the time key update queries.

We first show that Hung et al.’s scheme is vulnerable to Type I attacks.

**Type I Attacks on Hung et al.’s Scheme.**

Let $\mathcal{A}_I$ be a Type I adversary who knows the time update key and the secret value of any user. After getting a signature $(z, c)$ of a message $\mu$ for $(ID, t, PK_{ID})$ from a sign query, $\mathcal{A}_I$ intends to generate a new signature $(z', c')$ of the same message $\mu$ associated to $(ID, t', PK_{ID}')$, where $t'$ is a new time period and $PK_{ID}'$ is the replaced public key.

- **First**, $\mathcal{A}_I$ makes a sign query and gets a valid signature $$(z = (z_1, z_2, z_3, z_4, z_5, z_6), c)$$ on a message $\mu$ for $(ID, t, PK_{ID} = (P_{ID}, R_{ID}))$, where $$c = H_2(y_1 + h \ast y_2, y_3 + h \ast y_4, a_1 \ast y_5 + a_2 \ast y_6, \mu),$$ $$z_1 = y_1 + s_1 \ast c, \quad z_2 = y_2 + s_2 \ast c, \quad z_3 = y_3 + s_3 \ast c, \quad z_4 = y_4 + s_4 \ast c, \quad z_5 = y_5 + s_5 \ast c, \quad z_6 = y_6 + s_6 \ast c.$$ 
- **Next**, from the signature, $\mathcal{A}_I$ can compute $y_3, y_4, y_5$ and $y_6$ as $$z_3 - s_3 \ast c = y_3, \quad z_4 - s_4 \ast c = y_4, \quad z_5 - s_5 \ast c = y_5, \quad z_6 - s_6 \ast c = y_6,$$ since $\mathcal{A}_I$ knows the time update key $(s_3, s_4)$ such that $s_3 + h \ast s_4 = H_0(ID, t)$ and the user secret value $(s_5, s_6)$ such that $a_1 \ast s_5 + a_2 \ast s_6 = R_{ID}$.
- **After that**, $\mathcal{A}_I$ computes a new time update key $(s'_3, s'_4)$ satisfying $s'_3 + h \ast s'_4 = H_1(ID, t')$ corresponding to a new time period $t'$ and a secret value $(s'_5, s'_6)$ such that $a_1 \ast s'_5 + a_2 \ast s'_6 = R'_{ID}$, where the replaced public key is $PK_{ID}' = (P_{ID}, R'_{ID})$. Then $\mathcal{A}_I$ can compute $$z'_3 = y_3 + s'_3 \ast c, \quad z'_4 = y_4 + s'_4 \ast c,$$ from $(s'_3, s'_4), c$ and the computed values $(y_3, y_4)$ and $$z'_5 = y_5 + s'_5 \ast c, \quad z'_6 = y_6 + s'_6 \ast c,$$ from $(s'_5, s'_6), c$ and the computed values $(y_5, y_6)$.
- **Finally**, $\mathcal{A}_I$ outputs $$(z' = (z_1, z_2, z'_3, z'_4, z'_5, z'_6), c)$$ as a signature of the message $\mu$ associated to $(ID, t', PK_{ID}')$, where $PK_{ID}' = (P_{ID}, R'_{ID})$.
- **Then,** it is a valid certificateless signature on the message $\mu$ for $(ID, t', PK_{ID}')$ since it passes the verification
equation as
\[ c = H_2(z_1 + h \ast z_2 - H_0(ID) \ast c, z_3') \\
+ h \ast z_1' - H_1(ID, t') \ast c, \\
a_1 \ast z_3' + a_2 \ast z_6' - R'_d \ast c, \mu) \]
\[ = H_2(y_1 + h \ast y_2, y_3 + h \ast y_4, a_1 \ast y_5 + a_2 \ast y_6, \mu). \]
In fact, the above equation holds since the computed value in the inputs of the hash function in the two different signatures is the same as \((y_1 + h \ast y_2, y_3 + h \ast y_4, a_1 \ast y_5 + a_2 \ast y_6, \mu)\). Thus, the hashed value \(c\) in the two signatures is the same.

- Consequently, the adversary \(A_I\) succeeds in forging a certificateless signature of the same message \(\mu\) associated to \((ID, t', PK'_d)\) without using the partial private key related to \(ID\). It is a valid forgery in the Game of Type I adversary since \((ID, \mu, t')\) was never requested to the signing queries and \(ID\) was never requested to the private key extract queries.

Our attack is meaningful since signing on the same message for different time periods or different public keys can occur frequently. Our attack uses the fact that the value, \(y_1 + h \ast y_2, y_3 + h \ast y_4, a_1 \ast y_5 + a_2 \ast y_6\) and \(\mu\) being hashed are all the same in the signatures on the same message for different time periods or different public keys.

- The Type I adversary in our attack uses a signature \(z\) on \(\mu\) for \((ID, t, PK_d)\) for generating a new signature on the same message and the same identity for the different time period and different public key, \((ID, t', PK'_d)\).

- In the attack, despite the use of the different public key and the different time period, the values, \(y_1 + h \ast y_2, y_3 + h \ast y_4, a_1 \ast y_5 + a_2 \ast y_6\) and the message \(\mu\), are the same in the resulting signatures. In secure CLS schemes, the adversaries cannot forge valid signatures on \((\mu, ID, t, PK_d)\) from known signature on \((\mu, ID, t, PK_d)\).

2) ATTACKS ON XIE et al.’s CLS SCHEME
We present Type II attacks on Xie et al.’s scheme.

■ Type II Attacks on Xie et al.’s Scheme.
Let \(A_{II}\) be a Type II adversary with the master secret key \(msk\). After getting a signature \(sig\) of a message \(\mu\) for \((id, pk_{id})\), \(A_I\) intends to generate a new signature \(sig\) of the same message \(\mu\) for \((id, pk_{id})\), where \(id\) is a new identity.

- First, \(A_{II}\) makes a sign query and gets a valid signature \(\widehat{sig} = (e, z = (z_1, z_2, z_3, z_4))\) on a message \(\mu\) for \((id, pk_{id})\), where

\[ e = H_2(y_1 + h \ast y_2, y_3 + h \ast y_4, \mu), \]
\[ z_1 = y_1 + s_1 \ast e, z_2 = y_2 + s_2 \ast e, \]
\[ z_3' = y_3' + s_1' \ast e, z_4' = y_4' + s_2' \ast e. \]

- Next, \(A_{II}\) can compute
\[ z_1 - s_1 \ast e = y_1, z_2 - s_2 \ast e = y_2, \]

since \(A_{II}\) knows the partial private key \((s_1, s_2)\) such that \(s_1 + h \ast s_2 = H(id)\) from the master secret key \(msk\).

- After that, \(A_{II}\) with \(msk\) can compute a partial private key \((\hat{s}_1, \hat{s}_2)\) such that \(\hat{s}_1 + h \ast \hat{s}_2 = H(id)\) associated to a new identity \(\hat{id}\). Then \(A_{II}\) computes
\[ \hat{z}_1 = y_1 + \hat{s}_1 \ast e, \hat{z}_2 = y_2 + \hat{s}_2 \ast e, \]

from \((\hat{s}_1, \hat{s}_2), e\) and the computed values \((y_1, y_2)\).

- Finally, \(A_{II}\) outputs \(\widehat{sig} = (e, z = (\hat{z}_1, \hat{z}_2, \hat{z}_3, \hat{z}_4))\) as a signature of the message \(\mu\) for \((\hat{id}, pk_{id})\).

In the above equation holds since the computed values in the inputs of the hash function in the two different signatures are the same as \((y_1 + y_2 + h, y_3 + y_4, y_5 + y_6, \mu)\). The hashed value \(e\) in the two different signatures is the same.

- Consequently, \(A_{II}\) succeeds in forging a valid certificateless signature on the message \(\mu\) for \((\hat{id}, pk_{id})\) without the knowledge of the user secret key. It is a valid forgery in the Game of Type II adversary since \((\hat{id}, \mu)\) was never requested to the signing queries and \(\hat{id}\) was never requested to the secret value queries and the public key replacement queries.

III. VULNERABILITIES OF TWO CLS SCHEMES FROM PAIRINGS
Recently, Wang et al. [17] and Rezaeibagha et al. [11] constructed two CLS schemes from bilinear pairings. The security of their schemes was proven for Type I and Type II adversaries under the intractability of some mathematical problems. Here, we point out invalidity of Wang et al.’s scheme and vulnerability of Rezaeibagha et al.’s scheme against Type I attacks.

A. REVIEW OF TWO CLS SCHEMES FROM PAIRINGS
We first review Wang et al.’s aggregate certificateless signature (CLAS) scheme and Rezaeibagha et al.’s CLS scheme based on bilinear pairings.

1) WANG et al.’S CLS SCHEME
- Setup: Given a security parameter \(l\), KGC performs as follows:
- Choose an additive group \(G_1\) and a multiplicative group \(G_2\) with the same prime order \(q\) and \(P\) a generator of \(G_1\).
- Choose a bilinear pairing \(e : G_1 \times G_2 \rightarrow G_2\) and three cryptographic secure hash functions \(H_1 : \{0, 1\}^* \times G_1 \rightarrow G_1, H_2 : \{0, 1\}^* \times G_1 \rightarrow G_1^*\) and \(H_3 : \{0, 1\}^* \times G_1 \rightarrow G_1\).
Choose a master secret key $s$ and calculate $P_{pub} = sP$ as a master public key, where $s$ is used for partial key extraction.

Publish $< l, q, e, \mathbb{G}_1, \mathbb{G}_2, P, H_1, H_2, H_3, P_{pub} >$ as the system parameters and store the master key $s$ secretly.

DC generates its secret/public key as $(S_{DC}, PK_{DC})$, where $PK_{DC} = S_{DC} \cdot P$.

**UserRegistration:** This algorithm generates the client $C_i$’s public key and full private key.

- $C_i$ with an identity $ID_i$ chooses a secret key $S_{1i}$ randomly and calculates a public key $PK_{1i} = S_{1i} \cdot P$. Then $C_i$ sends $(ID_i, PK_{1i})$ to KGC via a secure channel.
- After receiving $(ID_i, PK_{1i})$ from $C_i$, KGC calculates $PK_{2i} = H_1(ID_i, PK_{1i})$ and $S_{2i} = sPK_{2i}$. Then KGC stores the tuple $(ID_i, PK_{1i}, PK_{2i}, S_{2i})$ and sends $(S_{2i}, PK_{2i})$ to $C_i$ through a secure way. After that $C_i$ sets $(S_{1i}, S_{2i})$ as a private key.

**Signing:** For healthsensing data $m_i$ and a private key $(S_{1i}, S_{2i})$, $C_i$ does the followings:

- Choose a random value $k_i$ and calculate $V_i = k_iP$, $h_i = H_2(m_i||t_i, V_i)$, $W_i = H_3(m_i||t_i, V_i)$, $U_i = h_iS_{12} + k_iP_{pub} + S_{1i}W_i$, $SN_i = ePK_{2i}(ID_i, W_i)$.
- Output $(U_i, V_i)$ as a signature of $m_i$, where $t_i$ is a current time stamp.
- Finally, $C_i$ sends $(U_i, V_i, SN_i, m_i, t_i)$ to DC.

**Verification:** On the receipt of $(U_i, V_i, SN_i, m_i, t_i)$ from $C_i$, if $t_i$ is invalid then DC rejects it. Otherwise, DC does the followings:

- DC first decrypts $SN_i$ by using $s_{DC}$ as $D_{DC}(SN_i) = ID_i||W_i$ and calculates $PK_{1i} = H_1(ID_i, PK_{1i})$ and $h_i = H_2(m_i||t_i, V_i)$.
- Check the equality $e(U_i, P) = e(V_i + h_iPK_{2i}, P_{pub})e(PK_{1i}, W_i)$.

If it holds, output valid.

**Aggregation:** After receiving $< (U_1, V_1, SN_1', m_1, t_1), \ldots, (U_n, V_n, SN_n', m_n, t_n) >$ from a set of identities $(ID_1)_{i=1}^n$ and a set of public keys $(PK_1)_{i=1}^n$, an aggregator calculates

$$U = \sum_{i=1}^n U_i, \quad V = \sum_{i=1}^n (V_i + PK_{2i}),$$

$$W = \sum_{i=1}^n W_i, \quad PK = \sum_{i=1}^n PK_{1i}.$$  

Then $(U, V, W, PK)$ is an aggregate authentication message on all health sensing data $(m_i)_{i=1}^n$.

**Aggregate Verification:** From an aggregate signature $(U, V, W, PK)$, DC checks the equality

$$e(U, P) = e(V, P_{pub})e(PK, W).$$

If it is valid, DC uploads the data.

2) REZAIEBAGHA et al.’s CLS SCHEME

Rezaiebagha et al. [11] proposed a new lightweight CLS scheme for Industrial Internet of Things (IIoT). They proved its unforgeability for the Type I and II adversaries under the hardness assumption of the q-BSDH problem. The q-BSDH problem is: given $g, g^s, g^{s \cdot t}, \ldots, g^{s^k}$, find $(c, e(g, g)^{\frac{1}{q}})$. Their scheme runs as follows:

- **Setup.** Setup($\lambda$). For a security parameter $\lambda$, KGC does the followings:
  
  - Choose $\mathbb{G}$ and $\mathbb{G}_T$ as the groups of prime order $p$ and a bilinear pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.
  - Choose two hash functions $H : \mathbb{Z}_p^* \rightarrow \mathbb{Z}_p$ and $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}$ and a master secret key $msk = s$ in $\mathbb{Z}_p^*$, and compute $Y_{KGC} = g^s$ as a master public key.
  - Output system parameters $\text{params} = (p, \mathbb{G}, \mathbb{G}_T, g, g_1, e, Y_{KGC}, H, H_1)$.

- **Set-Partial-Private-Key($params, msk, ID_i$).** From the master secret key $s$, $\text{params}$ and a user’s identity $ID_i$, and KGC calculates $d_i = g_1^i$, $Y_{1i} = g_i$, where $g_i = H_1(ID_i) \in \mathbb{G}$. Then $d_i$ is a partial private key for the user with $ID_i$. If it satisfies the following equations then the user accepts it,

$$e(d_i, g) = e(g_1, Y_{KGC}).$$

- **Set-Secret-Value($ID_i$).** From $\text{params}$ and a user’s identity $ID_i$, output a secret key $SK_i = x_i$ for a random $x_i \in \mathbb{Z}_p$.

- **Set-Public-Key($x_i, ID_i$).** From $\text{params}$, a user’s identity $ID_i$, the master secret key $s$ and the secret value $x_i$, output a user public key $Y_i = (Y_{1i}, Y_{2i})$, where $Y_{1i} = g_i$ and $Y_{2i} = g^{x_i}$.

- **CLS-Sign($params, SK_i, d_i, m$).** From $\text{params}$, the secret key $(SK_i, d_i)$ and a message $m \in \mathbb{Z}_p$, select $r \in \mathbb{Z}_p$, calculate

$$S = d_i^{r} = H_{12}(Y_{12}, m^{y_i})$$

and output $\sigma = (r, S)$ as a signature on $m$.

- **CLS-Verify($params, ID_i, Y_i, m, \sigma$).** From $\text{params}$, a user’s public key $Y_i = (Y_{1i}, Y_{2i})$ and a signature $\sigma = (r, S)$ on $m$, it checks the following equality

$$e(Y_{1i}, Y_{KGC}) = e(S, (g_1^{r} \cdot Y_{12}^{H(r, Y_{12}, m)})).$$

If it is valid, accept the signature.

### B. VULNERABILITIES OF TWO CLS SCHEMES FROM PAIRINGS

Now, we present vulnerability of Rezaiebagha et al.’s CLS scheme against Type I attacks and invalidity of Wang et al.’s CLS scheme.
1) INVALIDITY OF WANG et al.’s CLAS SCHEME
Wang et al. [17] claimed that their Aggregate Verification was correct as:
\[ e(U, P) = e(\prod_{i=1}^{n} U_i, P) = \prod_{i=1}^{n} e(h_iS_{i2} + k_iP_{pub} + S_{i1}W_i, P) \]
\[ = \prod_{i=1}^{n} e(h_iS_{i2} + k_iP_{pub}, P) \prod_{i=1}^{n} e(PK_{i1}, W_i) \]
\[ = e(V, P_{pub})e(PK, W). \]
However, the above equality is not correct since
\[ \prod_{i=1}^{n} e(PK_{i1}, W_i) \neq e(PK, W). \]
The calculation \( \prod_{i=1}^{n} e(h_iS_{i2} + k_iP_{pub}, P) = e(V, P_{pub}) \) is valid since
\[ \prod_{i=1}^{n} e(h_iS_{i2} + k_iP_{pub}, P) = \prod_{i=1}^{n} e(h_iPK_{i2} + k_iP, P_{pub}) \]
\[ = e(\sum_{i=1}^{n} [h_iPK_{i2} + V_i], P_{pub}) \]
\[ = = (V, P_{pub}). \]

\( e \) is bilinear map and \( P_{pub} \) is a fixed point. However, in the calculation of \( e(PK, W) \), we have
\[ e(PK, W) = e(\sum_{i=1}^{n} PK_{i1}, \sum_{i=1}^{n} W_i) \]
\[ = \prod_{i=1}^{n} e(PK_{i1}, W_i) = \prod_{i=1}^{n} e(PK_{i1}, W) \]
from the bilinearity of \( e \) on the left side or
\[ e(PK, W) = e(\sum_{i=1}^{n} PK_{i1}, \sum_{i=1}^{n} W_i), \]
\[ = \prod_{i=1}^{n} e(\sum_{i=1}^{n} PK_{i1}, W_i) = \prod_{i=1}^{n} e(PK, W_i) \]
from the bilinearity of \( e \) on the right side. Thus,
\[ e(PK, W) = \prod_{i=1}^{n} e(PK_{i1}, W) \neq \prod_{i=1}^{n} e(PK_{i1}, W_i), \]
\[ e(PK, W) = \prod_{i=1}^{n} e(PK, W_i) \neq \prod_{i=1}^{n} e(PK_{i1}, W_i). \]
The bilinearity of \( e \) cannot applied to both sides. Consequently, Wang et al.’s CLAS scheme cannot work.

2) TYPE I ATTACK ON REZAIEBAGHA et al.’s SCHEME
We show that Rezaiebagha et al.’s CLS scheme is vulnerable to Type I attacks, where the Type I adversary can replace the user public key and knows the secret key related to the public key.

**A Type I attack on Rezaiebagha et al.’s Scheme.**
Let \( \mathcal{A}_I \) be a Type I adversary. First, \( \mathcal{A}_I \) replaces \( Y_i \) with the public key \( Y_i \) of a user \( ID_i \). After getting a signature on a message \( m \) for the replaced public key \( Y_i' \), \( \mathcal{A}_I \) intends to forge a certificateless signature on a new message \( m' \neq m \) for the user \( ID_i \).

- \( \mathcal{A}_I \) selects a random secret value \( x' \in \mathbb{Z}_p^* \), calculates \( Y_{i2}' = g_1^{x'} \) and replaces a public key \( Y_i' = (Y_i = g_l, Y_{i2}' = g_1^{x'}) \) with \( Y_i = (Y_i = g_l, Y_{i2} = g_1^{x'}) \) corresponding to \( ID_i \), where \( g_l = H_1(ID_i) \). Thus, \( \mathcal{A}_I \) knows the secret value \( x' \).
- After replacing the public key, \( \mathcal{A}_I \) obtains a signature \( \sigma = (r, S) \) on \( m \) from the signing oracle, where
\[ S = d_i^{1/H_1(Y_{i2}' \cdot m)} = g_i^{r/H_1(Y_{i2}' \cdot m)}. \]
- Then, \( \mathcal{A}_I \) can obtain the partial private key \( g_i' \) by computing
\[ (S)^{r/H_1(Y_{i2}' \cdot m)x'} = g_i^{s_i}. \]
from the values \( x', r \) and \( S \).
- At last, \( \mathcal{A}_I \) who knows the value \( g_i' \) picks a random value \( r' \in \mathbb{Z}_p^* \) and generates a signature \( \sigma' = (r', S') \) on a new message \( m' \), where
\[ S' = (g_i')^{r'/(H_1(Y_{i2}' \cdot m'))} = g_i^s \]
Then it is valid signature since it passes the verification equation as
\[ e(S', (g_i)^{H_1(Y_{i2}' \cdot m')}) = e(g_i', g) \]
\[ = e(Y_{i1}', Y_{KG}) \]
Thus, \( \mathcal{A}_I \) can forge a valid certificateless signature on the message \( m' \neq m \) for \( \{ID_i, Y_i' = (Y_i, Y_{i2}') \} \).

In our attack, since the random value \( r \) in the signature is known, the adversary who knows the secret key \( x' \) can compute the partial private key \( g_i' \) corresponding to \( ID_i \) from the known signature. Therefore, the adversary who knows both the secret key \( x' \) and the partial private key \( g_i' \) can forge signatures on new messages. If the value \( r \) is hidden then our attack cannot work.

IV. SOME IMPROVEMENTS
A. IMPROVEMENTS ON THE CLS SCHEMES FROM NTRU LATTICES
Now, we suggest an improved version of the RCLS scheme to prevent our attacks. Our attacks are meaningful since signing on the same message at different time periods or different public keys can occur frequently. The vulnerabilities of the CLS schemes against our attacks are due to the fact that the values, \( y_1 + h \cdot y_2, y_3 + h \cdot y_4, a_1 + y_5 + a_2 \cdot y_6 \) and \( \mu \) being hashed are all the same although different time periods and different public keys used in the signing on the same message. They can be prevented by binding a signature with the public key, the identity and the current time period to the input being hashed. It can be achieved by simple modifications in Sign and Verify of Huang et al.’s RCLS scheme as:
Sign: Given a message $\mu \in \{0, 1\}^*$, a signer with the private key chooses random values $y_1, y_2, y_3, y_4, y_5, y_6$ by the distribution $D_{\mu}^N$ and computes $c = H_2(y_1 + h \cdot y_2, y_3 + h \cdot y_4, a_1 \cdot y_5 + a_2 \cdot y_6, \mu, ID, PK_{ID}, t)$. If it holds, output valid.

Verify: Given a signature $(z = (z_1, z_2, z_3, z_4, z_5, z_6), c)$ as a signature of the message $\mu$.

Then, output $(z = (z_1, z_2, z_3, z_4, z_5, z_6), c)$ as a signature of the message $\mu$.

If it holds, output valid.

Adding the related information $(ID, t, PK_{ID})$ into the input of the hash function $H_2$ plays an important role in binding a signature with the public key, the identity and the current time period. Thus, the hashed values for $(\mu, ID, t, PK_{ID})$ and $(\mu, ID, t', PK'_{ID})$ are all different, so our attack cannot work any more.

The same method can be applied to Xie et al.’s CLS scheme.

B. IMPROVEMENTS ON THE CLS SCHEMES FROM PAIRINGS

In our attack on Rezaeibagha et al.’s scheme, the known random value $r$ in the signature makes the partial private key $g_r^t$ recovered, thus, hiding $r$ can prevent our attack. It can be achieved by simple modifications in Sign and Verify of the scheme as:

- **CLS-Sign**(params, $SK_i$, $d_i$, $m$): From the public parameters $\text{params}$, the secret key $(SK_i, d_i)$ and a message $m \in Z_p$, it chooses $r \in Z_p$ and calculates $g^r$ and $S = d_i^{r \cdot H(g^t \cdot r'_{11}, r'_{12}, m_{11, 12})} = g_{i}^{r \cdot H(g^t \cdot r'_{11}, r'_{12}, m_{11, 12})}$.

Then $\sigma = (g^r, S)$ is a signature of $m$.

- **CLS-Verify**(params, $ID_i$, $Y_i$, $m$, $s$): From $\text{params}$, a user’s public key $Y_i = (Y_{i1}, Y_{i2})$ and a signature $\sigma = (g^r, S)$ on $m$, it checks the equality $e(Y_{i1}, Y_{KGC}) = e(S, g_{12}^{r \cdot H(g^t \cdot r'_{11}, r'_{12}, m)})$.

If it is valid, accept the signature.

V. CONCLUSION

We showed that Huang et al.’s RCLS scheme and Xie et al.’s CLS scheme were insecure against Type I or Type II attacks. Our results showed that their security proofs for existential unforgeability against the Type I or Type II adversaries were flawed. The lack of binding a signature with the identity, the public key and the current time period causes vulnerabilities of the schemes against Type I or Type II attacks. Adding the related information $(ID, t, PK_{ID})$ into the input of the hash function $H_2$ for signing can prevent our attacks without requiring additional computational cost and communication overhead. We also pointed out invalidity of Wang et al.’s CLAS scheme and vulnerability of Rezaeibagha et al.’s CLS scheme against Type I attacks.

REFERENCES

[1] F.-T. Zhang, Y.-X. Sun, L. Zhang, M.-M. Geng, and S.-J. Li, “Research on certificateless public key cryptography,” J. Softw., vol. 22, no. 6, pp. 1316–1332, Jun. 2011.

[2] L. Ducas, V. Lyubashevsky, and T. Prest, “Efficient identity-based encryption over NTRU lattices,” Adv. Cryptol., vol. 8874, pp. 22–41, Dec. 2014.

[3] C. Gentry, C. Peikert, and V. Vaikuntanathan, “Trapdoors for hard lattices and new cryptographic constructions,” in Proc. 14th Annu. ACM Symp. Theory Comput. (STOC), 2008, pp. 197–206.

[4] O. Goldreich, S. Goldwasser, and S. Halevi, “Public-key cryptosystems from lattice reduction problems,” Adv. Cryptol., vol. 1294, pp. 112–131, Aug. 1997.

[5] I. Hoffstein, J. Pipher, and H. Joseph Silverman, “NTRU: A ring-based public key cryptosystem,” ANT'S-III, vol. 1423, pp. 267–288, Jul. 1998.

[6] Y.-H. Hung, Y.-M. Tseng, and S.-S. Huang, “A revocable certificateless short signature scheme and its authentication application,” Informatica, vol. 27, no. 3, pp. 549–572, Sep. 2016.

[7] Y.-H. Hung, Y.-M. Tseng, and S.-S. Huang, “Lattice-based revocable certificateless signature,” Symmetry, vol. 9, no. 10, p. 242, 2017.

[8] V. Lyubashevsky and D. Micciancio, “Asymptotically efficient lattice-based digital signatures,” J. Cryptol., vol. 31, no. 3, pp. 774–797, Jul. 2018.

[9] V. Lyubashevsky, “Lattice signatures without trapdoors,” Adv. Cryptol., vol. 7237, pp. 738–755, Feb. 2012.

[10] P. Nguyen and O. Regev, “Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures,” J. Cryptol., vol. 22, pp. 139–160, Sep. 2009.

[11] F. Rezaeibagha, Y. Ma, X. Huang, W. Yang, and K. Huang, “Fully secure lightweight certificateless signature scheme for IoT,” IEEE Access, vol. 7, pp. 144433–144443, 2019.

[12] A. Shamir, “Identity-based cryptosystems and signature schemes,” Adv. Cryptogr., vol. 196, pp. 47–53, May 1984.

[13] M. Tian and L. Huang, “Efficient identity-based signature from lattices,” SEC, vol. 428, pp. 321–329, Jun. 2014.

[14] M. Tian and L. Huang, “Certificateless and certificate-based signatures from lattices,” Secur. Commun. Netw., vol. 8, no. 8, pp. 1575–1586, May 2015.

[15] M. Tian and L. Huang, “Identity-based signatures from lattices: Simpler, faster, Shorter*,” Fundamenta Informaticae, vol. 145, no. 2, pp. 171–187, Jun. 2016.

[16] T.-T. Tsai, S.-S. Huang, and Y.-M. Tseng, “Secure certificateless signature with revocation in the standard model,” Math. Problems Eng., vol. 2014, pp. 1–16, 2014.

[17] W. Wang, H. Huang, Y. Wu, and Q. Huang, “Cryptanalysis and improvement of an anonymous batch verification scheme for mobile healthcare crowd sensing,” IEEE Access, vol. 7, pp. 165842–165851, 2019.

[18] J. Xie, Y. Hu, J. Gao, W. Gao, and M. Jiang, “Efficient Certificateless Signature Scheme on NTRU Lattice,” KSII Trans. Interns. Inf. Syst. vol. 10, no. 10, pp. 5190–5208, 2016.