Hindawi
Security and Communication Networks
Volume 2018, Article ID 3578942, 11 pages
https://doi.org/10.1155/2018/3578942

Research Article

An Efficient Certificateless Generalized Signcryption Scheme

Bo Zhang,[1,2,3] **Zhongtian Jia**,[1,3] **and Chuan Zhao**[1]

[1]*School of Information Science and Engineering, University of Jinan, Jinan 250022, China*
[2]*School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia*
[3]*Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan 250022, China*

Correspondence should be addressed to Bo Zhang; zhangbosdu@gmail.com

Received 3 November 2017; Revised 2 March 2018; Accepted 5 April 2018; Published 15 May 2018

Academic Editor: Jiankun Hu

Generalized signcryption can adaptively work as an encryption scheme, a signature scheme, or a signcryption scheme with only one algorithm. The paper proposes an efficient certificateless generic signcryption scheme without utilizing bilinear pairing operations. It is proved to satisfy confidentiality and unforgeability against chosen ciphertext and message attacks in an adaptive manner, respectively, in the random oracle model. Due to the lower computational cost and communication overhead, the proposed scheme is suitable for low power and processor devices.

1. Introduction

In the traditional Public Key Infrastructure (PKI), a certificate authority (CA) which is a third party issues the certificates to bind the identity of a user and the corresponding public key. The certificate provides an unforgeable and trusted link by CA's digital signature. However, the problem of certificate management, including the storage, revocation, and distribution of certificates, is complex in this kind of PKI. Identity-based Public Key Cryptosystems (ID-PKC) were introduced by Shamir [1] in 1984 to simplify certificate management problem. A user's public key can be easily derived from arbitrary strings corresponding to his identity information, such as passport number, telephone number, name, and email address. A trusted third party named private key generator (PKG) computes private keys from a master secret and users' identity information and distributes these private keys to users participating in the scheme. This eliminates the need for certificates as used in a traditional PKI. ID-based systems may be a good alternative for certificate-based systems from the viewpoint of efficiency and convenience. But an inherent problem of ID-based cryptosystems is the key escrow; that is, the PKG knows the user's private key, resulting in no user privacy and authenticity. To eliminate these problems simultaneously, Al-Riyami and Paterson introduced the concept of certificateless public key cryptography (CL-PKC) in 2003 [2]. In a CL-PKC, a public/secret key pair is produced by the user himself independently without requiring the public key to be certified. Also, a partial private key is generated by a semi-trusted third party, called the key generation center (KGC), from the unique identifier information of the user. Knowing only one of them should not be able to impersonate the user and carry out any of the cryptographic operations as the user. In other words, CL-PKC can act as an intermediate between traditional PKI and ID-PKC.

The confidentiality and authenticity of messages are the basic requirement for secure communication. In 1997, Zheng proposed a cryptographic primitive signcryption [3], which simultaneously fulfils the integrated function of public encryption and digital signature with a computing and communication cost significantly smaller than that required by the signature-then-encryption method. According to the three public key authentication methods, signcryption can be divided into three types: PKI-based signcryption schemes [4–10], ID-based signcryption schemes (IBSC) [11–16], and certificateless signcryption schemes (CLSC) [17–20].

Sometimes, confidentiality and authenticity are needed separately, and sometimes, both of them are needed simultaneously. We can use encryption, signature, or signcryption to achieve the security properties, respectively. But maintaining three different primitives or components at the same time is quite a burden to a system, especially to the low

power and processor devices in low-bandwidth environments. Generalized signcryption (GSC) was proposed [21–23] to solve this problem. GSC scheme can adaptively work as encryption scheme, signature scheme, or signcryption scheme with only one algorithm. In other words, without any additional modification and computation, it provides double functions when confidentiality and authenticity are required simultaneously and the separate encryption or signature function when one of them is required. So, the GSC scheme can be viewed as a primitive with three work modes. In 2010, the first certificateless generalized signcryption (CLGSC) scheme was introduced by Ji et al. [24]. In their work, the formal definition, security model, and a concrete scheme were proposed. But Kushwah and Lai [25] noted that the scheme [24] is not existentially unforgeable against Type I adversary, and they proposed a new secure and efficient CLGSC scheme. Zhou et al. [26] proposed a more efficient CLGSC scheme based on the certificateless signcryption proposed in [17]. However, all the existing CLGSC schemes are realized with bilinear pairing operations. Compared with other operations, the bilinear pairing operation is much more complicated. Therefore, a concrete scheme without bilinear pairing is more suitable for applications. Very recently, Zhou et al. [27] introduced the key-insulated mechanism into GSC and propose a concrete scheme without bilinear pairings in the certificateless cryptosystem setting.

In this paper, we give a formal definition and the security concept of CLGSC and propose an efficient concrete scheme without utilizing bilinear pairing operations based on a certificateless signcryption-tag key encapsulation mechanism [28]. The concrete scheme is proved to satisfy confidentiality and unforgeability against chosen ciphertext and message attacks in an adaptive manner, respectively, in the random oracle model. Due to less computational cost and communication overhead, the proposed scheme is suitable for low power and processor devices.

The rest of the paper is organized as follows. The security problems, complexity assumptions, and the formal model of CLGSC scheme are introduced in Section 2. We describe a new CLGSC scheme in Section 3 and give the security proof and performance analysis of the new scheme in Sections 4 and 5, respectively. Finally, the conclusions are given in Section 6.

2. Preliminaries

2.1. Security Problems and Complexity Assumptions. Several related mathematical hard problems and security assumptions are presented here.

(i) The Elliptic Curve Discrete Log Problem (ECDLP) [29]: for group G_q which is generated by $P \in G_q$, given $Q \in_R G_q$, to find $x \in \mathbb{Z}_q^*$ such that $Q = xP$.

Definition 1 (ECDLP assumption). For group G_q which is generated by $P \in G_q$, given $Q \in_R G_q$, the successful advantage of any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ is presented as $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{ECDLP}} = \Pr[A(P, Q = xP) = x \mid x \in \mathbb{Z}_q^*]$. If there exists no PPT adversary $\mathcal{A}$ with nonnegligible advantage ϵ in solving the ECDLP problem, we say that the ECDLP assumption holds.

(ii) One-sided Gap Diffie-Hellman problem (ECDLP) [30]: for group G_q which is generated by $P \in G_q$, $Q = xP$ is a fixed point, given $R = yP$, to find xyP with the help of a one-sided decision Diffie-Hellman (ODDH) oracle. The ODDH oracle gets the tuple (P, xP, yP, cP) as the input and outputs 1 if $c = ab(\mathrm{mod}q)$ and 0 otherwise.

Definition 2 (ECDLP assumption). For group G_q which is generated by $P \in G_q$, $Q = xP$ is a fixed point, given $R = yP$. The successful advantage of any PPT adversary $\mathcal{A}$ is presented as $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{ECDLP}} = \Pr[A(P, xP, yP) = xyP \mid x, y \in \mathbb{Z}_q^*]$. If there exists no PPT adversary $\mathcal{A}$ with nonnegligible advantage ϵ by making q_o ODDH oracle queries in solving the ECDLP problem, we say that the (q_o, ϵ)-ECDLP assumption holds.

2.2. Certificateless Generic Signcryption Scheme (CLGSC)

2.2.1. Framework. Certificateless generic signcryption scheme (CLGSC) consists of the following probabilistic polynomial time algorithms.

(1) Setup. Take a security parameter l as input, KGC runs Setup algorithm to generate common parameters *params* and a master key *msk*. *params* are publicly available, whereas the *msk* is kept by the KGC secretly. Formally, we can write

$$(params, msk) \longleftarrow Setup\left(1^l\right). \tag{1}$$

(2) Set-User-Key. Take the common parameters *params* and the identity information *ID* of himself as input; each user runs Set-User-Key algorithm to generate a secure value and the corresponding public key value for himself. It returns the user's secret value x and a corresponding public value *PV*. Formally, we can write

$$(x, PV) \longleftarrow Set\text{-}User\text{-}Key\left(params, ID\right). \tag{2}$$

(3) Extract-Partial-Private-Key. Given the common parameters *params*, an identity *ID*, and the corresponding public value *PV*, KGC runs Extract-Partial-Private-Key algorithm to generate the partial private key d associated with *ID*. It distributes d to the user via a secure channel. Formally, we can write

$$d \longleftarrow Extract\text{-}partial\text{-}private\text{-}key\left(params, msk, ID, PV\right). \tag{3}$$

(4) Set-Private-Key. Given the common parameters *params*, the partial private key d, and the secret value x, the user with identity *ID* runs this algorithm to generate the full private key *SK* for himself. Formally, we can write

$$SK \longleftarrow Set\text{-}private\text{-}key\left(params, x, d\right). \tag{4}$$

(5) Set-Public-Key. Given the common parameters *params*, the partial private key d, the secret value x, and the public value *PV*, the user with identity *ID* runs this algorithm to

generate the full public key PK as the output. Formally, we can write

$$PK \longleftarrow \textit{Set-public-key}\left(params, x, d, PV\right). \quad (5)$$

(6) CLGSC-Signcrypt. Given the common parameters *params*, the message m, the receiver's identity ID_B, and the full public value PK_B, the user with identity ID_A and the full private key SK_A runs this algorithm to generate the ciphertext δ as the output. Note that ID_A and ID_B could be null string. Formally, we can write

$$\left(ID_A, ID_B, \delta\right) \longleftarrow \textit{CLGSC-signcrypt}\left(params, m, ID_A, SK_A, ID_B, PK_B\right). \quad (6)$$

(7) CLGSC-Unsigncrypt. Given the ciphertext δ, the sender's identity ID_A, and the public key PK_A, the receiver with identity ID_B and the full private key SK_B runs this algorithm to unsigncrypt (or decrypt) the ciphertext. It returns m or true for the valid signcryption ciphertext or signature; return $\perp$ means invalid. Note that ID_A and ID_B could be null string. Formally, we can write

$$\left(m/true/\perp\right) \longleftarrow \textit{CLGSC-unsigncrypt}\left(params, \delta, ID_A, PK_A, ID_B, SK_B\right). \quad (7)$$

2.2.2. Security Model. A CLGSC must satisfy confidentiality in encryption mode or signcryption mode and unforgeability in signcryption mode or signature mode. In a CLGSC scheme, we must consider two types of adversaries: a common user of the system and a honest-but-curious KGC. A common user cannot be in possession of the master secret key generated by KGC. But he can replace the public key of the users with valid public keys of his choice in an adaptive manner. This type of adversary is modeled by the Type I adversary. An honest-but-curious KGC knows the KGC's master secret key. But he is not able to replace the public keys of the users. This type of adversary is modeled by the Type II adversary.

An adversary $\mathcal{A}$ can access seven kinds of oracles as follows.

Set-User-Key Queries. $\mathcal{A}$ requests the secret value for a user U with ID_U. $\mathcal{C}$ uses the Set-User-Key algorithm to compute (x_u, PV_U) and sends x_U to $\mathcal{A}$. If U's public key has already been replaced, then a Type I adversary cannot submit U's identity ID_U and requests the secret value of U.

Extract-Partial-Private-Key Queries. $\mathcal{A}$ requests the partial private key for a user U with ID_U; $\mathcal{C}$ uses the Set-User-Key algorithm to compute (x_u, PV_U) and then sends a partial private key d_U generated by the Extract-Partial-Private-Key algorithm to $\mathcal{A}$.

Set-Private-Key Queries. $\mathcal{A}$ requests the private key for a user U with ID_U; $\mathcal{C}$ sends the full private key SK_U generated by the Set-User-Key algorithm and Extract-Partial-Private-Key algorithm to $\mathcal{A}$. Note that if U's public key has already been replaced, then a Type I adversary cannot submit the identity ID_U and requests the full private key of U.

Set-Public-Key Queries. $\mathcal{A}$ requests the public key for a user U with ID_U; $\mathcal{C}$ returns the public key PK_U to $\mathcal{A}$ generated by the Set-User-Key algorithm and Extract-Partial-Private-Key algorithm.

Public-Key-Replacement Queries. $\mathcal{A}$ computes a new public key PK'_U for ID_U and replaces PK_U. Note that a Type II adversary cannot access Public-Key-Replacement queries.

CLGSC-Signcrypt Queries. $\mathcal{A}$ submits (ID_A, ID_B, m) to $\mathcal{C}$, in which m is a message and ID_A and ID_B are the sender's and the receiver's identities, respectively. $\mathcal{C}$ returns the ciphertext δ to $\mathcal{A}$. Note that if the public key of the sender has been replaced, then $\mathcal{C}$ may not return the ciphertext δ. In this case, $\mathcal{A}$ must provide the secret value to $\mathcal{C}$.

CLGSC-Unsigncrypt Queries. $\mathcal{A}$ submits (ID_A, ID_B, δ) to $\mathcal{C}$, in which δ is a signature or signcryption ciphertext and ID_A and ID_B are the sender's and the receiver's identities, respectively. $\mathcal{C}$ returns the output of CLGSC-unsigncrypt to $\mathcal{A}$. Note that if the public key of the receiver is replaced, then $\mathcal{C}$ may not return the corresponding value. In this case, $\mathcal{A}$ must provide the secret value to $\mathcal{C}$.

Confidentiality

Definition 3 (IND-CLGSC-CCA2 confidentiality). A certificateless generic signcryption scheme in signcryption mode or encryption mode is semantically secure against adaptive chosen ciphertext attacks if, for all PPT adversary, the advantage is negligible in the following games. The games are played between a challenger $\mathcal{C}$ and the adversaries $\mathcal{A}_I$ and $\mathcal{A}_{II}$, respectively.

GAME 1 (IND-CLGSC-CCA2-I)

Initial. $\mathcal{C}$ generates the system parameters *params* and the master secret key *msk* by running the Setup algorithm. It keeps *msk* secret and sends *params* to $\mathcal{A}_I$.

Phase I. $\mathcal{A}_I$ performs a polynomially bounded number of the above queries.

Challenge. $\mathcal{A}_I$ outputs a tuple (m_0, m_1, ID_A, ID_B), in which m_0 and m_1 are distinct messages of equal length and ID_A and ID_B are the sender's and the receiver's identities, respectively. Here, it is to be noted that ID_B's full private key has not been extracted by $\mathcal{A}_I$ in Phase I. It is also to be noted that ID_B's partial private key has not been extracted and his public key has not been replaced simultaneously. $\mathcal{C}$ picks $\gamma \in \{0, 1\}$ randomly, runs the algorithm of CLGSC-signcrypt with (ID_A, ID_B, m_γ), and sends the output δ to $\mathcal{A}_I$.

Phase II. $\mathcal{A}_I$ asks queries adaptively again. However, the full private key for ID_B may not be extracted by $\mathcal{A}_I$ and the partial private key for ID_B may not be extracted if the public key of ID_B has been replaced in Phase I. Only after the public key

PK_A or PK_B has been replaced, CLGSC-unsigncrypt query on δ with sender ID_A and receiver ID_B is allowed.

Guess Stage. $\mathscr{A}_I$ outputs his guess γ' and if $\gamma' = \gamma$ he wins the game.

The advantage of $\mathscr{A}_I$ is $Adv_{\mathscr{A}_I}^{IND\text{-}CLGSC\text{-}CCA2\text{-}I} = 2\Pr[\gamma' = \gamma] - 1$.

GAME 2 (IND-CLGSC-CCA2-II)

Initial. $\mathscr{C}$ generates *params* and *msk* by running the Setup algorithm. It sends *params* and *msk* to $\mathscr{A}_{II}$.

Phase I. $\mathscr{A}_{II}$ performs a polynomially bounded number of queries just as $\mathscr{A}_I$ in IND-CLGSC-CCA2-*I* game. Extract-Partial-Private-Key queries are not included here, because $\mathscr{A}_{II}$ knows *msk*, and he can generate users' partial private keys by himself.

Challenge. At the end of Phase *I*, $\mathscr{A}_{II}$ outputs a tuple (m_0, m_1, ID_A, ID_B), in which m_0 and m_1 are distinct messages of equal length and ID_A and ID_B are the sender's and the receiver's identities, respectively. Here, it is to be noted that $\mathscr{A}_{II}$ must have made no Set-Private-Key queries on ID_B in Phase *I*. $\mathscr{C}$ picks $\gamma \in \{0, 1\}$ randomly, runs the algorithm of CLGSC-signcrypt with (m_γ, ID_A, ID_B), and sends the output δ to $\mathscr{A}_{II}$.

Phase II. $\mathscr{A}_{II}$ asks queries adaptively again. However, the full private key for ID_B may not be extracted and only after the public key PK_A or PK_B has been replaced, CLGSC-unsigncrypt query on δ with sender ID_A and receiver ID_B is allowed.

Guess Stage. $\mathscr{A}_{II}$ outputs his guess γ' and if $\gamma' = \gamma$ he wins the game.

The advantage of $\mathscr{A}_{II}$ is $Adv_{\mathscr{A}_{II}}^{IND\text{-}CLGSC\text{-}CCA2\text{-}II} = 2\Pr[\gamma' = \gamma] - 1$.

Note that, in the above games, only the signcryption mode and encryption mode of the CLGSC scheme must be considered. The receiver's identity ID_B cannot be vacant. If the sender's identity ID_A is not vacant, the algorithm runs in signcryption mode; otherwise it runs in encryption mode.

Unforgeability

Definition 4 (EUF-CLGSC-CMA unforgeability). A certificateless generic signcryption scheme in signature mode or signcryption mode is existentially unforgeable against adaptive chosen message attacks if, for all PPT adversary, the advantage is negligible in the following games. The games are played between a challenger $\mathscr{C}$ and the adversaries $\mathscr{F}_I$ and $\mathscr{F}_{II}$, respectively.

GAME 3 (EUF-CLGSC-CMA-I)

Initial. $\mathscr{C}$ generates *params* and *msk* by running the Setup algorithm. It keeps *msk* secret and sends *params* to $\mathscr{F}_I$.

Training Phase. Like $\mathscr{A}_I$ in Phase *I* of the IND-CLGSC-CCA2-*I* game, $\mathscr{F}_I$ may perform a series of adaptive queries.

Forgery. $\mathscr{F}_I$ outputs a tuple $(ID_A^*, ID_B^*, \delta^*)$. It must not be an output of the CLGSC-signcrypt query. The full private key of ID_A^* must not be extracted by $\mathscr{F}_I$ during the Training Phase. Moreover, $\mathscr{F}_I$ must have not replaced ID_A^*'s public key and extracted ID_A^*'s partial private key simultaneously. If the output of CLGSC-unsigncrypt is not $\perp$, $\mathscr{F}_I$ wins the game.

GAME 4 (EUF-CLGSC-CMA-II)

Initial. $\mathscr{C}$ generates *params* and *msk* by running the Setup algorithm. It sends *params* and *msk* to $\mathscr{F}_{II}$.

Training Phase. Like $\mathscr{A}_{II}$ in Phase *I* of the IND-CLGSC-CCA2-*II* game, $\mathscr{F}_{II}$ may perform a series of adaptive queries.

Forgery. $\mathscr{F}_{II}$ outputs a tuple $(ID_A^*, ID_B^*, \delta^*)$. It must not be an output of the CLGSC-signcrypt query. During the Training Phase, $\mathscr{F}_{II}$ must have made no Set-Private-Key queries and Set-User-Key queries on ID_A^*. If the output of CLGSC-unsigncrypt is not $\perp$, $\mathscr{F}_{II}$ wins the game.

Note that, in the above games, only the signcryption mode and signature mode of the CLGSC scheme must be considered. The sender's identity ID_A cannot be vacant. If the receiver's identity ID_B is not vacant, the algorithm runs in signcryption mode; otherwise it runs in signature mode.

3. The Concrete Scheme

Motivated by the pairing-free CLSC-TKEM protocol, in this section, we present a novel certificateless generalized signcryption scheme. It consists of seven algorithms.

(1) Setup. Given a security parameters $l \in \mathbb{Z}^+$, the KGC executes the following operations:

(i) It chooses a l-bits prime q and the tuple $\{F_q, E/F_q, G_q, P\}$, where G_q is generated by P.

(ii) It chooses $s \in_R \mathbb{Z}_q^*$ uniformly as the master key and computes $P_{pub} = s \cdot P$.

(iii) Let H_0, H_1, H_2 be cryptography hash functions, where $H_0 : \{0,1\}^* \times G_q^2 \to \mathbb{Z}_q^*$, $H_1 : G_q^3 \times \{0,1\}^* \to \{0,1\}^w$, $H_2 : \{0,1\}^* \times \{0,1\}^* \times G_q^3 \to \mathbb{Z}_q^*$, where w is the plaintext block length.

(iv) Define an index function $f(ID)$ as follows: $f(ID) = 0$ if $ID = \emptyset$; otherwise, $f(ID) = 1$.

(v) The public parameters and functions are presented as $params = \{F_q, E/F_q, G_q, P, P_{pub}, H_0, H_1, H_2\}$.

(2) Set-User-Key. A user **U** with the identity ID_U randomly chooses $x_U \in \mathbb{Z}_q^*$ as its secret value and computes the corresponding public value as $PV_U = x_U \cdot P$.

(3) Extract-Partial-Private-Key. **U** sends (ID_U, PV_U) to the KGC. In turn, the KGC generates and returns the partial private key of **U** as follows:

(i) It chooses $r_U \in \mathbb{Z}_q^*$ and computes $R_U = r_U \cdot P$.

(ii) It computes $d_U = r_U + s \cdot H_0(ID_U, R_U, PV_U) \bmod q$.

d_U is the partial private key of **U**. **U** can accept d_U as a valid partial private key by determining if $d_U \cdot P = R_U + H_0(ID_U, R_U, PV_U) \cdot P_{pub}$ holds.

(4) Set-Private-Key. The user **U** takes the pair (x_U, d_U) as its full private key SK_U.

(5) Set-Public-Key. The user **U** takes the pair (PV_U, R_U) as its full public key PK_U.

(6) CLGSC-Signcrypt. With the message m and the receiver's identity ID_B, the sender **A** performs as follows:

(i) It chooses $s_A \in \mathbb{Z}_q^*$ randomly and computes $V = s_A \cdot P$.

(ii) It computes $f(ID_A)$, $f(ID_B)$.

(iii) It computes $c = H_1(V, T, s_A PV_B, ID_B) \cdot f(ID_B) \oplus m$, where $T = s_A \cdot (H_0(ID_B, R_B, PV_B) \cdot P_{pub} + R_B + PV_B)$.

(iv) It computes $W = (d_A + x_A \cdot H_2(ID_A, m, V, T, PV_A) + s_A \cdot H_2(ID_A, m, V, T, R_A)) \cdot f(ID_A)$.

(v) It sets $\delta = (V, W, c)$ and returns (ID_A, ID_B, δ) as the ciphertext.

(7) CLGSC-Unsigncrypt. Given the ciphertext $(ID_A, ID_B, \delta = (V, W, c))$, the receiver ID_B decrypts and verifies the ciphertext as follows:

(i) It computes $f(ID_A)$, $f(ID_B)$.

(ii) It computes $m' = c \oplus (H_1(V, T', x_B V, ID_B) \cdot f(ID_B))$, where $T' = V \cdot (x_B + d_B)$.

(iii) It checks if $W \cdot P = f(ID_A)(R_A + H_0(ID_A, R_A, PV_A) \cdot P_{pub} + PV_A \cdot H_2(ID_A, m', V, T', PV_A) + V \cdot H_2(ID_A, m', V, T', R_A))$. If the equation does not hold, then return $\perp$ indicating the message is not valid. Otherwise, return true when $f(ID_B) = 0$ indicating it is a valid signature of user **A** or m indicating it is a valid encryption/signcryption ciphertext of the message m sent to user **B**.

Correctness of the Scheme. The correctness of the proposed concrete scheme is proved as follows.

(i) Correctness of the Encryption

$$\begin{aligned} m' &= c \oplus H_1\left(V, T', x_B V, ID_B\right) = c \oplus H_1\left(V, V\right. \\ &\cdot \left(x_B + d_B\right), x_B s_A P, ID_B\Big) = c \oplus H_1\left(V, s_A\right. \\ &\cdot \left(x_B \cdot P + d_B \cdot P\right), s_A x_B P, ID_B\Big) = c \oplus H_1\left(V, s_A\right. \\ &\cdot \left(PV_B + R_B + H_0\left(ID_B, R_B, PV_B\right) \cdot P_{pub}\right), s_A PV_B, \\ &ID_B\Big) = c \oplus H_1\left(V, T, s_A PV_B, ID_B\right) = H_1\left(V, T,\right. \\ &\left. s_A PV_B, ID_B\right) \oplus m \oplus H_1\left(V, T, s_A PV_B, ID_B\right) = m. \end{aligned} \tag{8}$$

(ii) Correctness of the Signature

$$\begin{aligned} W \cdot P &= \left(d_A + x_A \cdot H_2\left(ID_A, m, V, T, PV_A\right) + s_A\right. \\ &\cdot H_2\left(ID_A, m, V, T, R_A\right)\Big) \cdot P = d_A \cdot P + x_A \cdot P \\ &\cdot H_2\left(ID_A, m, V, T, PV_A\right) + s_A \cdot P \\ &\cdot H_2\left(ID_A, m, V, T, R_A\right) = r_A \cdot P + s \cdot P \\ &\cdot H_0\left(ID_A, R_A, PV_A\right) + x_A \cdot P \\ &\cdot H_2\left(ID_A, m, V, T, PV_A\right) + s_A \cdot P \\ &\cdot H_2\left(ID_A, m, V, T, R_A\right) = R_A + H_0\left(ID_A, R_A, PV_A\right) \\ &\cdot P_{pub} + PV_A \cdot H_2\left(ID_A, m, V, T, PV_A\right) + V \\ &\cdot H_2\left(ID_A, m, V, T, R_A\right). \end{aligned} \tag{9}$$

4. Security Analysis of the Proposed Scheme

In this section, the security of the proposed concrete CLGSC scheme is proved as follows.

4.1. Confidentiality

Theorem 5. *The CLGSC scheme is semantically secure against adaptive chosen ciphertext attacks in encryption mode or signcryption mode in the random oracle model.*

Theorem 5 is proved based on Lemmas 6 and 7.

Lemma 6. *If an adversary $\mathcal{A}_I$ has a nonnegligible advantage ε against the IND-CLGSC-CCA2-I security of our scheme and performing q_{H_i} queries to oracles H_i $(i = 0, 1, 2)$, q_{ppk} Extract-Partial-Private-Key queries, and q_{sk} Set-Private-Key queries, then there is an algorithm that solves the ECDLP problem with probability $\varepsilon \cdot (1 - q_{ppk}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{ppk} - q_{sk})) \cdot 1/q_{H_1}$.*

Proof. Given an instance of the ECDLP problem, for group G_q generated by P and a fixed second point $Q(= aP)$ having as input $R(= bP) \in G_q$, $\mathcal{C}$ has to compute for the point $S(= cP) \in G_q$ such that $c = ab(\bmod q)$ with the help of a ODDH oracle. Suppose the IND-CLGSC-CCA2-*I* security of the CLGSC can be violated by a Type *I* adversary $\mathcal{A}_I$. $\mathcal{C}$ can utilize $\mathcal{A}_I$ to compute abP as the solution to this instance by the following interactive game.

$\mathcal{C}$ chooses $s \in_R \mathbb{Z}_q^*$ uniformly as the master key and computes $P_{pub} = s \cdot P$. It sends *params* to $\mathcal{A}_I$ and maintains lists L_i $(0 \le i \le 2)$ to keep the consistency between the responses to the hash queries and a list L_k of issued keys which are initially empty. $\mathcal{C}$ selects t randomly, where $1 \le t \le q_{H_0}$, and takes ID_t as the target identity. $\mathcal{C}$ chooses $e_t, x_t \in_R \mathbb{Z}_q^*$ and sets $H_0(ID_t, R_t, PV_t) = -e_t$, $R_t = e_t P_{pub} + aP - x_t P$, and $PV_t = x_t P$. $\mathcal{C}$ inserts $(ID_t, R_t, PV_t, -e_t)$ into the list L_0 and $(ID_t, \perp, x_t, R_t, PV_t)$ into the list L_k.

$\mathcal{C}$ answers $\mathcal{A}_I$'s queries to random oracles H_i $(0 \le i \le 2)$ as follows.

(i) H_0 *queries*: when $\mathscr{A}_I$ submits a H_0 query with (ID_i, R_i, PV_i) for some $i \in [1, q_0]$, $\mathscr{C}$ checks in L_0, and if $(ID_i, R_i, PV_i, -e_i)$ exists, $\mathscr{C}$ returns $-e_i$. Otherwise, $\mathscr{C}$ chooses $e_i \in_R \mathbb{Z}_q^*$ and returns $-e_i$ to $\mathscr{A}_I$. Then $\mathscr{C}$ inserts $(ID_i, R_i, PV_i, -e_i)$ into the list L_0.

(ii) H_1 *queries*: when $\mathscr{A}_I$ submits a H_1 query with (V_i, T_i, Y_i, ID_i) for some $i \in [1, q_1]$, $\mathscr{C}$ sets the tuple (aP, V_i, T_i) as the input of ODDH orale. If the output of ODDH oracle is 1, then $\mathscr{C}$ returns T_i as the solution abP and stops; else $\mathscr{C}$ searches in L_1, if $(V_i, *, Y_i, ID_i, l_i)$ exists, it replaces the symbol $*$ with T_i and returns l_i. Otherwise, $\mathscr{C}$ chooses $l_i \in_R \{0,1\}^n$ and returns l_i to $\mathscr{A}_I$. Then $\mathscr{C}$ inserts $(V_i, T_i, Y_i, ID_i, l_i)$ into the list L_1.

(iii) H_2 *queries*: when $\mathscr{A}_I$ submits a H_2 query with $(ID_i, m_i, V_i, T_i, PV_i/R_i)$ for some $i \in [1, q_2]$, $\mathscr{C}$ checks in L_2, and if $(ID_i, m_i, V_i, T_i, PV_i/R_i, h_i)$ exists in L_2, $\mathscr{C}$ returns h_i. Otherwise, $\mathscr{C}$ chooses $h_i \in_R \mathbb{Z}_q^*$ and returns h_i to $\mathscr{A}_I$. Then $\mathscr{C}$ inserts $(ID_i, m_i, V_i, T_i, PV_i/R_i, h_i)$ into L_2.

$\mathscr{C}$ can answer $\mathscr{A}_I$'s other queries as follows.

Phase I

(i) Set-User-Key Queries. $\mathscr{A}_I$ requests a secret value of the user with ID_i. If the public key of ID_i has not been replaced, then $\mathscr{C}$ responds with x_i by retrieving from the list L_k.

(ii) Extract-Partial-Private-Key Queries. $\mathscr{A}_I$ requests the partial private key of a user with ID_i. If $ID_i = ID_t$, $\mathscr{C}$ aborts the execution. Otherwise, $\mathscr{C}$ checks in L_k, and if $(ID_i, d_i, x_i, R_i, PV_i)$ exists, $\mathscr{C}$ returns d_i. Otherwise, $\mathscr{C}$ computes the partial private key of ID_i by using the actual Extract-Partial-Private-Key algorithm, and $\mathscr{C}$ inserts $(ID_i, d_i, x_t, R_i, PV_i)$ into the list L_k and returns d_i.

(iii) Set-Private-Key Queries. $\mathscr{A}_I$ requests a user's full private key with ID_i. $\mathscr{C}$ aborts the execution when $ID_i = ID_t$. Otherwise, $\mathscr{C}$ checks in L_k, and if $(ID_i, d_i, x_i, R_i, PV_i)$ exists, $\mathscr{C}$ returns the corresponding private key (x_i, d_i). Otherwise, $\mathscr{C}$ picks $e_i, b_i, x_i \in_R \mathbb{Z}_q^*$, then sets $H_0(ID_i, R_i, PV_i) = -e_i$, $R_i = e_i P_{pub} + b_i P$, and computes $PV_i = x_i P \cdot d_i = b_i$ and it satisfies the equation $d_i P = R_i + H_0(ID_i, R_i, PV_i) P_{pub}$. $\mathscr{C}$ returns (x_i, d_i) to $\mathscr{A}_I$ and inserts $(ID_i, R_i, PV_i, -e_i)$ into the list L_0 and $(ID_i, d_i, x_i, R_i, PV_i)$ into L_k.

(iv) Set-Public-Key Queries. $\mathscr{A}_I$ requests a user's public key with ID_i. $\mathscr{C}$ checks in L_k, and if $(ID_i, d_i, x_i, R_i, PV_i)$ exists, $\mathscr{C}$ returns the corresponding public key (R_i, PV_i). Otherwise, $\mathscr{C}$ picks $e_i, b_i, x_i \in_R \mathbb{Z}_q^*$, then sets $H_0(ID_i, R_i, PV_i) = -e_i$, $R_i = e_i P_{pub} + b_i P$, and computes the public key as $PV_i = x_i P \cdot d_i = b_i$ and it satisfies the equation $d_i P = R_i + H_0(ID_i, R_i, PV_i) P_{pub}$. $\mathscr{C}$ returns (PV_i, R_i) to $\mathscr{A}_I$ and inserts $(ID_i, R_i, PV_i, -e_i)$ into the list L_0 and $(ID_i, d_i, x_i, R_i, PV_i)$ into L_k.

(v) Public-Key-Replacement Queries. $\mathscr{A}_I$ requests to replace a user's public key (R_i, PV_i) with chosen values (ID_i, R_i', PV_i'). $\mathscr{C}$ updates corresponding tuple with $(ID_i, -, -, R_i', PV_i')$.

(vi) CLGSC-Signcrypt Queries. $\mathscr{A}_I$ sends the tuple $(ID_A, PV_A, R_A, ID_B, PV_B, R_B, m)$ to $\mathscr{C}$. For each query (ID_A, ID_B), if $ID_A \neq ID_t$, $\mathscr{C}$ executes the Set-Private-Key algorithm to compute SK_A corresponding to ID_A. Then, $\mathscr{C}$ gets the ciphertext δ by running the actual CLGSC-signcrypt algorithm. $\mathscr{C}$ sends δ to $\mathscr{A}_I$. If $ID_A = ID_t$ (and hence, $ID_B \neq ID_t$), $\mathscr{C}$ can obtain the full private key SK_B corresponding to ID_B. $\mathscr{C}$ computes $V = rP - h_t^{-1} \cdot ap$, $T = V \cdot (x_B + d_B)$, sets $H_2(ID_A, m, V, T, PV_A) = h_t'$, $H_2(ID_A, m, V, T, R_A) = h_t$, and adds the tuples $(ID_A, m, V, T, PV_A, h_t')$ and $(ID_A, m, V, T, R_A, h_t)$ to the list L_2 in which $r, h_t, h_t' \in_R \mathbb{Z}_q^*$. $\mathscr{C}$ computes $c = H_1(V, T, s_B V, ID_B) \cdot f(ID_B) \oplus m$ and $W = r \cdot h_t + x_A \cdot h_t' - x_A$. $\mathscr{C}$ outputs $(ID_A, ID_B, \delta = (V, W, c))$ as the ciphertext.

The tuple $(ID_A, ID_B, \delta = (V, W, c))$ can pass the verification as the valid ciphertext because the equality holds as follows:

$$\begin{aligned} & R_A + H_0(ID_A, R_A, PV_A) \cdot P_{pub} + PV_A \\ & \quad \cdot H_2(ID_A, m, V, T, PV_A) + V \\ & \quad \cdot H_2(ID_A, m, V, T, R_A) = e_t P_{pub} + aP - x_t P \\ & \quad + (-e_t) P_{pub} + x_t P \cdot h_t' + \left(rP - h_t^{-1} aP\right) \cdot h_t \\ & \quad = \left(r \cdot h_t + x_t h_t' - x_t\right) \cdot P = W \cdot P. \end{aligned} \tag{10}$$

(vii) CLGSC-Unsigncrypt Queries. $\mathscr{A}_I$ submits (ID_A, ID_B, δ) to $\mathscr{C}$. If $ID_B \neq ID_t$, $\mathscr{C}$ obtains the receiver's private key and returns the output of CLGSC-unsigncrypt algorithm to $\mathscr{A}_I$. Note that if the receiver's public value is replaced, $\mathscr{C}$ may not obtain the receiver's secret value. In this case, receiver's secret value is requested to be provided by $\mathscr{A}$. Otherwise, $\mathscr{C}$ searches in L_2 for $(ID_A, m, V, T, PV_A, h_i)$ and $(ID_A, m, V, T, R_A, h_i')$. If the entries exist and the equality $W \cdot P = R_A + H_0(ID_A, R_A, PV_A) \cdot P_{pub} + PV_A \cdot h_i + V \cdot h_i'$ holds, T is retrieved. If $\mathscr{C}$ can find a tuple (V, T, Y, ID_B, l) in L_1 making the ODDH oracle return 1 when queries are on (ap, V, T), then the message is $c \oplus l$.

Challenge. $\mathscr{A}_I$ submits $(m_0, m_1, ID_A^*, ID_B^*)$ in which m_0 and m_1 are distinct messages of equal length and ID_A^* and ID_B^* are the sender's and the receiver's identities, respectively. Here, it is to be noted that $\mathscr{A}_I$ must have made no Set-Private-Key queries on ID_B^* in Phase *I*. It is also to be noted that ID_B^*'s partial private key has not been extracted and his public key has not been replaced simultaneously. $\mathscr{C}$ aborts the game if $ID_B^* \neq ID_t$. Otherwise, $\mathscr{C}$ generates the challenge ciphertext as follows.

(1) It sets $V^* = bP$ and chooses $T^* \in_R G_q$.

(2) It selects randomly a bit $\gamma \in \{0,1\}$ and a random hash value h and sets $c^* = m_\gamma \oplus h$.

(3) It selects W^*, satisfies the equation $W^* P = R_A + H_0(ID_A, R_A, PV_A) \cdot P_{pub} + PV_A \cdot H_2(ID_A, m, V^*, T^*, PV_A) + V^* \cdot H_2(ID_A, m, V^*, T^*, R_A)$, and sends $\delta^* = (V^*, W^*, c^*)$ to $\mathscr{A}_I$.

Phase II. $\mathscr{A}_I$ asks queries adaptively again. In addition, the full private key for ID_B^* may not be extracted by $\mathscr{A}_I$ and the partial

private key for ID_B^* may not be extracted if the public key of ID_B^* has been replaced in Phase *I*. Only after the public key PK_A^* or PK_B^* has been replaced, CLGSC-unsigncrypt query on δ with sender ID_A^* and receiver ID_B^* is allowed.

Guess. Since $\mathscr{A}_I$ is able to break the IND-CLGSC-CCA2-*I* security of the CLGSC, a H_1 query with (V^*, T^*, Y^*, ID_B^*) should have been asked. Note that $T^* = b \cdot (H_0(ID_B, R_B, PV_B) \cdot P_{pub} + R_B + PV_B) = abP$. Therefore, one of the T's in L_1 is the ECDLP problem's solution. $\mathscr{C}$ chooses one T randomly and outputs it as the solution.

In the above challenge query, the senders ID_A and ID_A^* can be $\emptyset$ for the encryption mode; otherwise, it works as signcryption. Thus, the proof is suitable for the two modes.

Analysis. Lets E_1, E_2, and E_3 be the events when $\mathscr{C}$ aborts this game.

(i) E_1 is an event in which the target identity ID_t's partial private key is queried by $\mathscr{A}_I$. The probability of E_1 is $\Pr[E_1] = q_{ppk}/q_{H_0}$.

(ii) E_2 is an event in which the target identity ID_t's private key is queried by $\mathscr{A}_I$. The probability of E_2 is $\Pr[E_2] = q_{sk}/q_{H_0}$.

(iii) E_3 is an event in which the target identity ID_t has not been chosen as the receiver by $\mathscr{A}_I$ during the challenge phase. The probability of E_3 is $\Pr[E_3] = 1 - 1/(q_{H_0} - q_{ppk} - q_{sk})$.

Thus, $\mathscr{C}$ does not abort this game's probability of $\Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3] = (1 - q_{ppk}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{ppk} - q_{sk}))$.

$\mathscr{C}$ chooses the solution of ECDLP problem from L_1's probability of $1/q_{H_1}$. So, the successful advantage of $\mathscr{C}$ is $ADV_{\mathscr{C}}^{OGDH} = \Pr[\mathscr{C}(P, aP, bP) = abP] = \varepsilon \cdot (1 - q_{ppk}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{ppk} - q_{sk})) \cdot 1/q_{H_1}$. □

Lemma 7. *If $\mathscr{A}_{II}$ has nonnegligible advantage ε against the IND-CLGSC-CCA2-II security of our scheme and performing q_{sv} Extract-Secret-Value queries, q_{sk} Set-Private-Key queries, and q_{H_i} queries to oracles H_i $(i = 0, 1, 2)$, then there is an algorithm that solves the ECDLP problem with probability $\varepsilon \cdot (1 - q_{sv}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{sk} - q_{sv}))$.*

Proof. Given an instance of the ECDLP problem, for group G_q generated by P and a fixed second point $Q(= aP)$ having as input $R(= bP) \in G_q$, $\mathscr{C}$ has to compute for the point $S(= cP) \in G_q$ such that $c = ab(\mathrm{mod}\, q)$ with the help of a ODDH oracle. Suppose the IND-CLGSC-CCA2-*II* security of the CLGSC can be violated by a Type *II* adversary $\mathscr{A}_{II}$. $\mathscr{C}$ can utilize $\mathscr{A}_{II}$ to compute abP as the solution to this instance by the following interactive game.

$\mathscr{C}$ chooses $s \in_R \mathbb{Z}_q^*$ uniformly as the master key and computes $P_{pub} = s \cdot P$. $\mathscr{C}$ sends *params* and s to $\mathscr{A}_{II}$ and maintains lists L_i $(0 \le i \le 2)$ to avoid the inconsistency between the responses to the hash queries and a list L_k of issued keys which is initially empty. $\mathscr{C}$ selects t randomly, where $1 \le t \le q_{H_0}$, and fixes ID_t as the target identity. $\mathscr{C}$ chooses $a_t, l_t \in_R \mathbb{Z}_q^*$, sets $H_0(ID_t, R_t, PV_t) = l_t$, and computes $R_t = a_t P, d_t = a_t + l_t \cdot s$, and the public key as $PV_t = aP$. $\mathscr{C}$ inserts (ID_t, R_t, PV_t, l_t) into the list L_0 and $(ID_t, d_t, \perp, R_t, PV_t)$ into the list L_k.

$\mathscr{C}$ answers $\mathscr{A}_{II}$'s queries to random oracles H_i $(0 \le i \le 2)$ as follows:

(i) H_0 *queries*: when $\mathscr{A}_{II}$ submits a H_0 query with (ID_i, R_i, PV_i) for some $i \in [1, q_0]$, $\mathscr{C}$ checks if there exists a tuple (ID_i, R_i, PV_i, l_i) in L_0. If such a tuple exists, $\mathscr{C}$ answers with l_i. Otherwise, $\mathscr{C}$ chooses $l_i \in_R \mathbb{Z}_q^*$ and returns l_i as the answer. Then $\mathscr{C}$ inserts (ID_i, R_i, PV_i, l_i) into the list L_0.

(ii) H_1 *queries*: when $\mathscr{A}_{II}$ submits a H_1 query with (V_i, T_i, Y_i, ID_i), where $i \in [1, q_1]$, $\mathscr{C}$ sets the tuple (aP, V_i, Y_i) as the input of ODDH oracle. If the output of ODDH oracle is 1, then $\mathscr{C}$ outputs Y_i as the solution; else $\mathscr{C}$ searches L_1 with entries $(V_i, T_i, *, ID_i, l_i)$. If such a tuple exists, it replaces the symbol $*$ with Y_i and returns l_i. Otherwise, $\mathscr{C}$ chooses $l_i \in_R \{0, 1\}^n$, inserts $(V_i, T_i, Y_i, ID_i, l_i)$ into L_1, and returns l_i to $\mathscr{A}_{II}$.

(iii) H_2 *queries*: when $\mathscr{A}_{II}$ submits a H_2 query with $(ID_i, m_i, V_i, T_i, PV_i/R_i)$, where $i \in [1, q_2]$, $\mathscr{C}$ checks whether $(ID_i, m_i, V_i, T_i, PV_i/R_i, h_i)$ exists in L_2. If it exists, $\mathscr{C}$ returns h_i. Otherwise, $\mathscr{C}$ chooses $h_i \in_R \mathbb{Z}_q^*$, inserts $(ID_i, m_i, V_i, T_i, PV_i/R_i, h_i)$ into L_2, and returns h_i.

$\mathscr{C}$ can answer $\mathscr{A}_{II}$'s other queries as follows.

Phase I

(i) Set-User-Key Queries. $\mathscr{A}_{II}$ requests a user's secret value with ID_i. If $ID_i = ID_t$, $\mathscr{C}$ aborts. If $ID_i \ne ID_t$, $\mathscr{C}$ checks for a tuple $(ID_i, d_i, x_i, R_i, PV_i)$ in L_k. If it exists, $\mathscr{C}$ returns x_i. Otherwise, $\mathscr{C}$ chooses $a_i, x_i, l_i \in_R \mathbb{Z}_q^*$, then sets $H_0(ID_i, R_i, PV_i) = l_i$, and computes $R_i = a_i P$, $d_i = a_i + l_i \cdot s$, and the public key as $PV_i = x_i P$. $\mathscr{C}$ inserts (ID_i, R_i, PV_i, l_i) into the list L_0 and $(ID_i, d_i, x_i, R_i, PV_i)$ into the list L_k and returns x_i.

(ii) Set-Private-Key Queries. $\mathscr{A}_{II}$ produces ID_i to $\mathscr{C}$ and requests a user's private key with ID_i. If $ID_i = ID_t$, $\mathscr{C}$ aborts. Otherwise, $\mathscr{C}$ checks for a tuple $(ID_i, d_i, x_i, R_i, PV_i)$ in L_k. If it exists, $\mathscr{C}$ returns (x_i, d_i). Otherwise, $\mathscr{C}$ chooses $a_i, x_i, l_i \in_R \mathbb{Z}_q^*$, then sets $H_0(ID_i, R_i, PV_i) = l_i$, and computes $R_i = a_i P$, $d_i = a_i + l_i \cdot s$, and $PV_i = x_i P$. $\mathscr{C}$ inserts (ID_i, R_i, PV_i, l_i) into the list L_0 and $(ID_i, d_i, x_i, R_i, PV_i)$ into the list L_k and returns (x_i, d_i).

(iii) Set-Public-Key Queries. $\mathscr{A}_{II}$ requests a user's public key with ID_i. $\mathscr{C}$ checks for a tuple $(ID_i, d_i, x_i, R_i, PV_i)$. If it exists, $\mathscr{C}$ returns the corresponding public key (PV_i, R_i). Otherwise, $\mathscr{C}$ chooses $a_i, x_i, l_i \in_R \mathbb{Z}_q^*$, then sets $H_0(ID_i, R_i, PV_i) = l_i$, and computes $R_i = a_i P$, $d_i = a_i + l_i \cdot s$, and the public key as $PV_i = x_i P$. $\mathscr{C}$ inserts (ID_i, R_i, PV_i, l_i) into the list L_0 and $(ID_i, d_i, x_i, R_i, PV_i)$ into the list L_k and returns (PV_i, R_i).

(iv) CLGSC-Signcrypt Queries. $\mathscr{A}_{II}$ sends the tuple ID_A, $PV_A, R_A, ID_B, PV_B, R_B, m$ to $\mathscr{C}$. For each query (ID_A, ID_B), if $ID_A \ne ID_t$, $\mathscr{C}$ executes the Set-Private-Key algorithm to compute SK_A corresponding to ID_A. Then, $\mathscr{C}$ gets the ciphertext δ by running the actual CLGSC-signcrypt algorithm. $\mathscr{C}$ sends δ to $\mathscr{A}_{II}$. If $ID_A = ID_t$ (and hence, $ID_B \ne ID_t$), $\mathscr{C}$ can obtain the full private key SK_B corresponding to ID_B. $\mathscr{C}$ computes $V = rP - h_t^{-1} h_t' \cdot ap$,

$T = V \cdot (x_B + d_B)$, sets $H_2(ID_A, m, V, T, PV_A) = h_t'$, $H_2(ID_A, m, V, T, R_A) = h_t$, and adds the tuples $(ID_A, m, V, T, PV_A, h_t')$ and $(ID_A, m, V, T, R_A, h_t)$ to the list L_2 in which $r, h_t, h_t' \in_R \mathbb{Z}_q^*$. $\mathscr{C}$ computes $c = H_1(V, T, s_B V, ID_B) \cdot f(ID_B) \oplus m$ and $W = r \cdot h_t + d_t$. $\mathscr{C}$ outputs $(ID_A, ID_B, \delta = (V, W, c))$ as the ciphertext.

The tuple $(ID_A, ID_B, \delta = (V, W, c))$ can pass the verification as the valid ciphertext because the equality holds as follows:

$$\begin{aligned} & R_A + H_0(ID_A, R_A, PV_A) \cdot P_{pub} + PV_A \\ & \cdot H_2(ID_A, m, V, T, PV_A) + V \\ & \cdot H_2(ID_A, m, V, T, R_A) = a_t P + l_t P + aPh_t' \\ & + \left(rP - h_t^{-1} aP\right) \cdot h_t = \left(r \cdot h_t + a_t + sl_t\right) \cdot P \\ & = \left(r \cdot h_t + d_t\right) \cdot P = W \cdot P. \end{aligned} \tag{11}$$

(v) CLGSC-Unsigncrypt Queries. $\mathscr{A}_{II}$ submits (ID_A, ID_B, δ) to $\mathscr{C}$. If $ID_B \neq ID_t$, $\mathscr{C}$ obtains the receiver's private key, runs the CLGSC-unsigncrypt algorithm, and returns the output of CLGSC-unsigncrypt to $\mathscr{A}_{II}$. Otherwise, $\mathscr{C}$ searches in L_2 for $(ID_A, m, V, T, PV_A, h_i')$ and $(ID_A, m, V, T, R_A, h_i)$. If the entries exist and the equality $W \cdot P = R_A + H_0(ID_A, R_A, PV_A) \cdot P_{pub} + PV_A \cdot h_i' + V \cdot h_i$ holds, T is retrieved. If $\mathscr{C}$ can find a tuple (V, T, Y, ID_B, l) in L_1 making the ODDH oracle return 1 when queries are on (aP, V, Y), then the message is $c \oplus l$.

Challenge. $\mathscr{A}_{II}$ submits $(m_0, m_1, ID_A^*, ID_B^*)$ in which m_0 and m_1 are distinct messages of equal length and ID_A^* and ID_B^* are the sender's and the receiver's identities, respectively. Here, it is to be noted that $\mathscr{A}_{II}$ must have made no Set-Private-Key queries on ID_B^* in Phase *I*. $\mathscr{C}$ aborts the game if $ID_B^* \neq ID_t$. Otherwise, $\mathscr{C}$ generates the challenge ciphertext as follows.

(1) It sets $V^* = bP$, where bP is given in the instance of the ECDLP problem and computes $T^* = (x_B^* + d_B^*) \cdot V^*$.

(2) It selects randomly a bit $\gamma \in \{0, 1\}$ and h as a random hash value and sets $c^* = m_\gamma \oplus h$.

(3) It selects W^*, satisfies the equation $W^* P = R_A + H_0(ID_A, R_A, PV_A) \cdot P_{pub} + PV_A \cdot H_2(ID_A, m, V^*, T^*, PV_A) + V^* \cdot H_2(ID_A, m, V^*, T^*, R_A)$, and sends $\delta^* = (V^*, W^*, c^*)$ to $\mathscr{A}_{II}$.

Phase II. $\mathscr{A}_{II}$ asks queries adaptively again. In addition, it cannot query CLGSC-unsigncrypt on δ^*.

Guess. Since $\mathscr{A}_{II}$ is able to break the IND-CLGSC-CCA2-*II* security of the CLGSC, a H_1 query with (V^*, T^*, Y^*, ID_B^*) should have been asked. Note that $Y^* = x_B^* \cdot V^* = abP$. Therefore, one of the q_{H_1} values of Y^* in L_1 is the ECDLP problem's solution. $\mathscr{C}$ chooses one Y^* randomly and outputs it as the solution.

In the above challenge query, the senders ID_A and ID_A^* can be $\emptyset$ for the encryption mode; otherwise, it works as signcryption. Thus, the proof is suitable for the two modes.

Analysis. In order to assess the probability of success of the challenger, $\mathscr{C}$ lets E_1, E_2, and E_3 be the events in which $\mathscr{C}$ aborts the IND-CLGSC-CCA2-*II* game.

(i) E_1 is an event in which $\mathscr{A}_{II}$ asks to query the secret value of the target identity ID_t. The probability of E_1 is $\Pr[E_1] = q_{sv}/q_{H_0}$.

(ii) E_2 is an event in which $\mathscr{A}_{II}$ asks to query the private key of the target identity ID_t. The probability of E_2 is $\Pr[E_2] = q_{sk}/q_{H_0}$.

(iii) E_3 is an event in which the target identity ID_t has not been chosen as the receiver by $\mathscr{A}_{II}$ during the challenge. The probability of E_3 is $\Pr[E_3] = 1 - 1/(q_{H_0} - q_{sk} - q_{sv})$.

Thus, $\mathscr{C}$ does not abort the IND-CLGSC-CCA2-II game's probability of $\Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3] = (1 - q_{sv}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{sk} - q_{sv}))$.

So, the successful advantage of $\mathscr{C}$ is $ADV_{\mathscr{C}}^{OGDH} = \Pr[\mathscr{C}(P, aP, bP) = abP] = \varepsilon \cdot (1 - q_{sv}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{sk} - q_{sv}))$. □

4.2. Unforgeability

Theorem 8. *The CLGSC scheme in signcryption mode or signature mode is existentially unforgeable.*

Theorem 8 is proved based on Lemmas 9 and 10.

Lemma 9. *If an adversary $\mathscr{F}_I$ has nonnegligible advantage ε against the EUF-CLGSC-CMA-I security of our scheme and performing q_{ppk} Extract-Partial-Private-Key queries, q_{sk} Set-Private-Key queries, and q_{H_i} queries to oracles H_i $(i = 0, 1, 2)$, then there is an algorithm that solves the ECDLP problem with probability $\varepsilon \cdot (1 - q_{ppk}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{ppk} - q_{sk}))$.*

Proof. Given an instance of the ECDLP problem $(P, bP) \in G_q$, $\mathscr{C}$ must find b. Suppose the EUF-CLGSC-CMA-*I* security of the CLGSC can be violated by a forger $\mathscr{F}_I$. $\mathscr{C}$ can utilize $\mathscr{F}_I$ to compute b as the solution by the following interactive game.

$\mathscr{C}$ chooses $s \in_R \mathbb{Z}_q^*$ uniformly as the master key and computes $P_{pub} = sP$; it sends *params* to $\mathscr{F}_I$ and maintains lists L_i $(0 \leq i \leq 2)$ to keep consistency between the responses to the hash queries and a list L_k of issued keys which is initially empty.

Training Phase. $\mathscr{F}_I$ may make a series of queries and all of the queries are responded to identically as those queries in the IND-CLGSC-CCA2-*I* game.

Forgery. $\mathscr{F}_I$ returns a valid ciphertext from the sender ID_A to the receiver ID_B. If $ID_A \neq ID_t$, $\mathscr{C}$ aborts the execution of this game. We are ready to apply the forking lemma [31] that essentially says the following: consider the concrete scheme producing signatures (m, V, h, W) or signcryption ciphertexts of the form (c, V, h, W), where each of V, h, W corresponds to one of the three moves of a honest-verifier zero-knowledge protocol. If $\mathscr{F}_I$ is a sufficiently efficient forger in the above

interaction and forges a signature or signcryption ciphertext (V, W, c) in a time t with probability $\epsilon \geqslant 10(q_{sc}+1)(q_{sc}+q_h)/2^l$ (l being a security parameter so that h is uniformly taken from a set of 2^l elements) when making q_{sc} CLGSC-signcrypt queries and q_h random oracle calls and if the triples (V, h, W) can be simulated without knowing the private key, then there exists a Turing machine F' that uses $\mathscr{F}_I$ to produce two valid ciphertexts (V^*, W, c^*) and (V^*, W^*, c^*) on the same message m, in expected time $t' \leqslant 120686q_h t/\epsilon$. Thus, we can get $W \cdot P = R_A - e_t \cdot P_{pub} + h_t' \cdot PV_A + h_t \cdot V$ and $W^* \cdot P = R_A - e_t \cdot P_{pub} + h_t' \cdot PV_A + h_t^* \cdot V$. Let $V = bP$. We can obtain the following value.

$$\begin{aligned} W \cdot P - W^* \cdot P &= h_t \dot{V} - h_t^* \cdot V \\ &\Longrightarrow (W - W^*) \cdot P = (h_t - h_t^*) \cdot V \\ &\Longrightarrow (W - W^*) \cdot P = (h_t - h_t^*) \cdot bP \\ &\Longrightarrow W - W^* = (h_t - h_t^*) \cdot b \\ &\Longrightarrow (W - W^*)\,\dot{(}h_t - h_t^*)^{-1} = b. \end{aligned} \tag{12}$$

Therefore, $\mathscr{C}$ solve the ECDLP as $b = (W - W^*)\dot{(}h_t - h_t^*)^{-1}$ for the ECDLP problem.

In the above forgery query, the receiver ID_B can be $\emptyset$ for the signature mode; otherwise it works as signcryption. Thus, the proof is suitable for the two modes.

Analysis. Let E_1, E_2, and E_3 be the events when $\mathscr{C}$ aborts the game.

(i) E_1 is an event in which the target identity ID_t's partial private key is queried by $\mathscr{F}_I$. The probability of E_1 is $\Pr[E_1] = q_{ppk}/q_{H_0}$.

(ii) E_2 is an event in which the target identity ID_t's private key is queried by $\mathscr{F}_I$. The probability of E_2 is $\Pr[E_2] = q_{sk}/q_{H_0}$.

(iii) E_3 is an event in which the target identity ID_t has not been chosen as the sender by $\mathscr{F}_I$ during the forgery. The probability of E_3 is $\Pr[E_3] = 1 - 1/(q_{H_0} - q_{ppk} - q_{sk})$.

Thus, $\mathscr{C}$ does not abort the EUF-CLGSC-CMA-I game's probability of $\Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3] = (1 - q_{ppk}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{ppk} - q_{sk}))$.

So, the successful advantage of $\mathscr{C}$ is $ADV_{\mathscr{C}}^{ELDLP} = \Pr[\mathscr{C}(P, bP) = b] = \varepsilon \cdot (1 - q_{ppk}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{ppk} - q_{sk}))$. □

Lemma 10. *If an adversary $\mathscr{F}_{II}$ has nonnegligible advantage ε against the EUF-CLGSC-CMA-II security of our scheme performing q_{sv} Extract-Secret-Value queries and q_{H_i} queries to oracles H_i $(i = 0, 1, 2)$, then there is an algorithm $\mathscr{C}$ that solves the ECDLP problem with probability $\varepsilon \cdot (1 - q_{sv}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{sk} - q_{sv}))$.*

Proof. Given an instance of the ECDLP problem (P, bP), $\mathscr{C}$ must find b. Suppose the EUF-CLGSC-CMA-*II* security of the CLGSC can be violated by a forger $\mathscr{F}_{II}$. $\mathscr{C}$ can utilize $\mathscr{F}_{II}$ to compute b as the solution by the following interactive game.

$\mathscr{C}$ chooses $s \in_R \mathbb{Z}_q^*$ uniformly as the master key and computes $P_{pub} = sP$; it sends *params* and s to $\mathscr{F}_{II}$ and maintains lists L_i $(0 \leq i \leq 2)$ to keep consistency between the responses to the hash queries and a list L_k of issued keys which is initially empty.

Training Phase. $\mathscr{F}_{II}$ may make a series of queries and all the queries are responded to identically as those queries in the IND-CLGSC-CCA2-*II* game.

Forgery. Eventually, $\mathscr{F}_{II}$ returns a valid ciphertext from the sender ID_A to the receiver ID_B. If $ID_A \neq ID_t$, $\mathscr{C}$ aborts the execution of this game. It follows from the forking lemma [31] that if $\mathscr{F}_{II}$ is a sufficiently efficient forger in the above interaction, then we can construct another probabilistic polynomial time Turing machine $\mathscr{F}_{II}'$ that outputs two ciphertexts (V^*, W, c^*) and (V^*, W^*, c^*) on the same message m. Thus, we can get $W \cdot P = R_t + l_t \cdot P_{pub} + h_t' \cdot PV_t + h_t V$ and $W^* \cdot P = R_t + l_t \cdot P_{pub} + h_t' \cdot PV_t + h_t^* V$. Let $V = bP$. We can obtain the following value.

$$\begin{aligned} W \cdot P - W^* \cdot P &= h_t \dot{V} - h_t^* \cdot V \\ &\Longrightarrow (W - W^*) \cdot P = (h_t - h_t^*) \cdot V \\ &\Longrightarrow (W - W^*) \cdot P = (h_t - h_t^*) \cdot bP \\ &\Longrightarrow W - W^* = (h_t - h_t^*) \cdot b \\ &\Longrightarrow (W - W^*)\,\dot{(}h_t - h_t^*)^{-1} = b. \end{aligned} \tag{13}$$

Therefore, $\mathscr{C}$ solves the ECDLP as $b = (W - W^*)\dot{(}h_t - h_t^*)^{-1}$ for the ECDLP problem.

In the above forgery query, the receiver ID_B can be $\emptyset$ for the signature mode; otherwise it works as signcryption. Thus, the proof is suitable for the two modes.

Analysis. Let E_1, E_2, and E_3 be the events when $\mathscr{C}$ aborts the game.

(i) E_1 is an event in which the target identity ID_t's secret value is queried by $\mathscr{F}_{II}$. The probability of E_1 is $\Pr[E_1] = q_{sv}/q_{H_0}$.

(ii) E_2 is an event in which the target identity ID_t's private key is queried by $\mathscr{F}_{II}$. The probability of E_2 is $\Pr[E_1] = q_{sk}/q_{H_0}$.

(iii) E_3 is an event in which the target identity ID_t has not been chosen as the sender by $\mathscr{F}_{II}$ during forgery. The probability of E_3 is $\Pr[E_3] = 1 - 1/(q_{H_0} - q_{sk} - q_{sv})$.

Thus, $\mathscr{C}$ does not abort the EUF-CLGSC-CMA-II game's probability of $\Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3] = (1 - q_{sv}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{sk} - q_{sv}))$.

So, the successful advantage of $\mathscr{C}$ is $ADV_{\mathscr{C}}^{OGDH} = \Pr[\mathscr{C}(P, aP, bP) = abP] = \varepsilon \cdot (1 - q_{sv}/q_{H_0}) \cdot (1 - q_{sk}/q_{H_0}) \cdot (1/(q_{H_0} - q_{sk} - q_{sv}))$. □

5. Performance Analysis

Since computation time and ciphertext size are two important factors affecting efficiency, we compare our scheme

Table 1: Comparisons of the computational overhead and storage costs.

Schemes	Ciphertext size	CLGSC-signcrypt time	CLGSC-unsigncrypt time
[24]	$2\lvert G_1\rvert + \lvert m\rvert + \lvert ID\rvert + \lvert G_2\rvert + \lvert q\rvert$	$(3T_E + 2T_M + 4T_H) \approx 188.5\,T_{\text{Mul}}$	$(T_E + T_M + 4T_H + 2T_P) \approx 246.5\,T_{\text{Mul}}$
[25]	$2\lvert G_1\rvert + \lvert m\rvert + \lvert ID\rvert + \lvert G_2\rvert$	$(2T_E + 3T_M + 3T_H) \approx 174\,T_{\text{Mul}}$	$(T_E + 3T_M + 3T_H + 2T_P) \approx 304.5\,T_{\text{Mul}}$
[26]	$2\lvert G_1\rvert + \lvert m\rvert$	$(T_E + 4T_M + 3T_H + T_P) \approx 246.5\,T_{\text{Mul}}$	$(T_M + 3T_H + 5T_P) \approx 464\,T_{\text{Mul}}$
[27]*	$2\lvert G_1\rvert + \lvert q\rvert + \lvert m\rvert$	$7T_M + 7T_H \approx 203T_{\text{Mul}}$	$8T_M + 7T_H \approx 232\,T_{\text{Mul}}$
Ours	$2\lvert G_q\rvert + \lvert m\rvert$	$(5T_M + 4T_H) \approx 145\,T_{\text{Mul}}$	$(4T_M + 4T_H) \approx 116\,T_{\text{Mul}}$

*Note that the authors also considered the private key exposure problem in their paper.

with several existing schemes in these two terms from two aspects: CLGSC-signcrypt and CLGSC-unsigncrypt. We pay attention to operations such as bilinear pairing operations, exponentiation operations, scalar multiplication operations, and hash operations. We define the notations in the Notations section and adopt the experiment testing results from [32–34].

The comparison is shown in Table 1; $|G_1|$ denotes the size of an element in G_1, $|G_2|$ denotes the size of an element in G_2, $|m|$ denotes the length of message m, $|ID|$ denotes the length of identity ID, and $|q|$ is the size of an element in $\mathbb{Z}_q^*$.

Since the pairing and exponentiation operations require much more time than the multiplication operation, our proposed scheme is implemented without pairing and exponentiation operations. From Table 1, it shows that our proposed CLGSC scheme requires much less computational time than the other four schemes. So, our scheme has the shortest ciphertext size and is of high efficiency too.

6. Conclusion

In this paper, a concrete CLGSC scheme without utilizing bilinear pairing operations is proposed, and its security is proved in the random oracle model under the ECDLP and ECDLP assumptions, including security against both an adaptively chosen ciphertext attack and an existential forgery of Type *I* and *II* adversaries. The new scheme is computationally efficient and is suitable for low power and processor devices.

Notations

T_{Mul}: Time required for executing a modular multiplication operation
T_E: Time required for executing an exponentiation $T_E \approx 43.5T_{\text{Mul}}$
T_M: Time required for executing a scalar multiplication $T_M \approx 29T_{\text{Mul}}$
T_H: Time complexity for executing the simple hash function, which is negligible
T_P: Time required for executing a bilinear pairing operation $T_P \approx 87T_{\text{Mul}}$.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Nature Science Foundation of China (no. 61702218), Shandong Provincial Natural Science Foundation (no. ZR2014FL011, no. ZR2015FL023), and International Joint Training Program for Young and Middle-Aged Scholar of Shandong Province.

References

[1] A. Shamir, "Identity-based cryptosystem and signature scheme," in *Proceedings of the Cryptology-CRYPTO*, vol. 196 of *Lecture Notes in Computer Science*, pp. 120–126, 1984.

[2] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proceedings ofthe Proceedings of Cryptology-ASIACRYPT*, vol. 2894 of *Lecture Notes in Computer Science*, pp. 452–474, 2003.

[3] Y. L. Zheng, "Digital signcryption or how to achieve cost (signature & encryption) ≪ cost (Signature) + cost (encryption)," in *Proceedings of the Cryptology-CRYPTO*, vol. 1294 of *Lecture Notes in Computer Science*, pp. 165–179, 1997.

[4] Y. L. Zheng, "Signcryption and its applications in efficient public key solutions," in *Proceedings of the Information Security Workshop (ISW '97)*, vol. 1397 of *An invited lecture*, pp. 291–312, 1997.

[5] F. Bao and R. H. Deng, "A signcryption scheme with signature directly verifiable by public key," in *Proceedings of Public Key Cryptography*, vol. 1431 of *Lecture Notes in Computer Science*, pp. 55–59, 1998.

[6] C. Gamage, J. Leiwo, and Y. Zheng, "Encrypted message authentication by firewalls," in *Proceedings of the Public Key Cryptography*, vol. 1560 of *Lecture Notes in Computer Science*, pp. 69–81, 1999.

[7] J. Malone-Lee and W. Mao, "Two birds one stone: signcryption using RSA," in *Proceedings of the Cryptology-CT-RSA*, vol. 2612 of *Lecture Notes in Computer Science*, pp. 211–226, 2003.

[8] B. Libert and J. Quisquater, "Efficient Signcryption with Key Privacy from Gap Diffie- Hellman Groups," in *Proceedings of the Public Key Cryptography*, vol. 2947 of *Lecture Notes in Computer Science*, pp. 187–200, 2004.

[9] F. Li, H. Zhang, and T. Takagi, "Efficient signcryption for heterogeneous systems," *IEEE Systems Journal*, vol. 7, no. 3, pp. 420–429, 2013.

[10] F. Li, B. Liu, and J. Hong, "An efficient signcryption for data access control in cloud computing," *Computing*, vol. 99, no. 5, pp. 465–479, 2017.

[11] J. Malone-Lee, "Identity based signcryption," http://eprint.iacr.org.

[12] B. Libert and J. J. Quisquater, "A new identity based signcryption scheme from pairings," in *Proceedings of the IEEE Information Theory Workshop, ITW '03*, pp. 155–158, 2003.

[13] X. Boyen, "Multipurpose identity-based signcryption: a Swiss Army knife for identity-based cryptography," in *Proceedings of the Cryptology-Crypto*, vol. 2729 of *Lecture Notes in Computer Science*, pp. 383–399, 2003.

[14] L. Chen and J. Malone-Lee, "Improved identity-based signcryption," in *Proceedings of the Public Key Cryptography*, vol. 3386 of *Lecture Notes in Computer Science*, pp. 362–379, 2005.

[15] P. S. Barreto, B. Libert, N. McCullagh, and J. J. Quisquater, "Efficient and provably-secure identity-based signatures and signcryption from bilinear maps," in *Proceedings of the Cryptology-ASIACRYPT 2005*, vol. 3788 of *Lecture Notes in Computer Science*, pp. 515–532, 2005.

[16] H. J. Jo, J. H. Paik, and D. H. Lee, "Efficient privacy-preserving authentication in wireless mobile networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 7, pp. 1469–1481, 2014.

[17] M. Barbosa and P. Farshim, "Certificateless signcryption," in *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS '08)*, pp. 369–372, ACM, Tokyo, Japan, March 2008.

[18] F. G. Li, M. Shirase, and T. Takagi, "Certificateless hybrid signcryption," in *Proceedings of the ISPEC '09*, vol. 5451 of *Lecture Notes in Computer Science*, pp. 112–123, 2009.

[19] A. Yin and H. Liang, "Certificateless hybrid signcryption scheme for secure communication of wireless sensor networks," *Wireless Personal Communications*, vol. 80, no. 3, pp. 1049–1062, 2015.

[20] F. Li, Y. Han, and C. Jin, "Certificateless online/offline signcryption for the Internet of Things," *Wireless Networks*, vol. 23, no. 1, pp. 145–158, 2017.

[21] Y. L. Han and X. Y. Yang, "New ECDSA-verifiable generalized signcryption," *Chinese Journal of Computers*, vol. 29, no. 11, pp. 2003–2012, 2006.

[22] Y. L. Han, X. Y. Yang, P. Wei et al., "ECGSC: elliptic curve based generalized signcryption," in *Proceedings of the 3rd International Conference on Ubiquitous Intelligence and Computing- UIC '06*, vol. 4159 of *Lecture Notes in Computer Science*, pp. 956–965, 2006.

[23] Y. Han, "Generalization of signcryption for resources-constrained environments," *Wireless Communications and Mobile Computing*, vol. 7, no. 7, pp. 919–931, 2007.

[24] H. F. Ji, W. B. Han, and L. Zhao, "Certificateless generalized signcryption," Cryptology ePrint Archive, Report 2010/204, http://eprint.iacr.org.

[25] P. Kushwah and S. Lai, "Efficient generalized signcryption schemes," Cryptology ePrint Archive, Report 2010/346, http://eprint.iacr.org.

[26] C. X. Zhou, W. Zhou, and X. W. Dong, "Provable certificateless generalized signcryption scheme," *Designs, Codes and Cryptography*, vol. 71, no. 2, pp. 331–346, 2014.

[27] C. X. Zhou, Z. Zhao, W. Zhou et al., "Certificateless key-insulated generalized signcryption scheme without bilinear pairings," *Security and Communication Networks*, vol. 2017, Article ID 8405879, 17 pages, 2017.

[28] S.-H. Seo, J. Won, and E. Bertino, "pCLSC-TKEM: a pairing-free certificateless signcryption- tag key encapsulation mechanism for a privacy-preserving IoT," *Transactions on Data Privacy*, vol. 9, no. 2, pp. 101–130, 2016.

[29] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[30] N. Koblitz and A. Menezes, "Intractable problems in cryptography," in *Proceedings of the 9th International Conference on Finite Field and Their Applications, Contemporary Mathematics*, pp. 279–300, 2010.

[31] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, vol. 13, no. 3, pp. 361–396, 2000.

[32] Y. F. Chung, K. H. Huang, F. Lai, and T. S. Chen, "ID-based digital signature scheme on the elliptic curve cryptosystem," *Computer Standards & Interfaces*, vol. 29, no. 6, pp. 601–604, 2007.

[33] S. K. H. Islam and G. P. Biswas, "A pairing-free identity-based authenticated group key agreement protocol for imbalanced mobile networks," *Annals of Telecommunications-Annales des Télécommunications*, vol. 67, no. 11-12, pp. 547–558, 2012.

[34] 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.