Des. Codes Cryptogr. (2014) 71:331–346
DOI 10.1007/s10623-012-9734-y

Provable certificateless generalized signcryption scheme

Caixue Zhou · Wan Zhou · Xiwei Dong

Received: 1 December 2011 / Revised: 6 April 2012 / Accepted: 23 July 2012 /
Published online: 3 August 2012

Abstract Generalized signcryption can adaptively work as an encryption scheme, a signature scheme or a signcryption scheme with only one algorithm. It is very suitable for storage-constrained environments. In this paper, we introduce a formal security model for certificateless generalized signcryption schemes secure against the malicious-but-passive key generation center attacks and propose a novel scheme. Our scheme is proved to be IND-CCA2 secure under the GBDH assumption and CDH assumption and existentially unforgeable under the GDH' assumption and CDH assumption in random oracle model. Furthermore, performance analysis shows the proposed scheme is efficient and practical.

Keywords Certificateless generalized signcryption · Malicious-but-passive KGC attacks · Gap bilinear Diffie–Hellman assumption · Computational Diffie–Hellman assumption · Random oracle model

Mathematics Subject Classification (2010) 94A60

1 Introduction

The notion of identity based (ID-based) cryptosystem was introduced by Shamir [1] in 1984 as an approach to simplify public key and certificate management in a public key infrastructure (PKI), but the key escrow problem of ID-based cryptosystem is inherent. To avoid this problem, Al-Riyami and Paterson [2] proposed a new cryptographic primitive as certificateless public key system in 2003.

Communicated by C. Blundo.

C. Zhou (✉) · W. Zhou · X. Dong
School of Information Science and Technology, University of Jiujiang, Jiujiang 332005, Jiangxi, People's Republic of China
e-mail: charlesjjjx@126.com

In 1997, Zheng [3] proposed a novel conception named signcryption. The purpose of signcryption is to perform encryption and signature simultaneously more efficiently than the sign-then-encrypt approach. In 2008, Barbosa and Farshim [4] extended the concept of signcryption to certificateless cryptographic system.

Signcryption plays a great role in some network environments when confidentiality and authenticity are needed simultaneously. Now consider the scenarios where sometimes we need confidentiality and authenticity separately and sometimes, we need both simultaneously. To achieve this, we can use three different schemes: an encryption scheme, a signature scheme and a signcryption scheme. But, in the low bandwidth environment e.g. smartcards and wireless sensor networks (WSN) etc., we cannot afford to use three different schemes to achieve confidentiality and authenticity separately or simultaneously. Motivated by this, Han et al. [5,6] in 2006 proposed a new primitive called generalized signcryption (GSC), which can adaptively work as an encryption scheme, a signature scheme or a signcryption scheme with only one algorithm, meanwhile they gave a GSC scheme based on ECDSA [7]. In 2007, Wang et al. [8] gave the first security model and improved the scheme proposed in [5]. The first ID-GSC scheme along with a security model was proposed by Lal and Kushwah [9] in 2008. However in 2010, Yu et al. [10] showed that the security model proposed in [9] is not complete, they improved the security model and proposed a concrete scheme which is secure in the new model. In 2011, Kushwah and Lal [11] simplified the security model proposed in [10] and gave an efficient ID-GSC scheme. In addition, many other GSC schemes [12–17] have been proposed too, which include multi-receiver schemes [12,13], multi-PKG scheme [17], schemes in the standard model [16,17].

Ji et al. [18] first introduced the notion of certificateless generalized signcryption (CLGSC) in 2010. They gave the formal definition of CLGSC and its security model, and proposed a concrete scheme. But Kushwah and Lai [19] pointed out scheme [18] is not secure and they proposed a new secure and efficient CLGSC scheme. In the same year, Ji et al. [20] proposed another CLGSC scheme based on scheme [4], however, [21] gave an attack on scheme [4], so scheme [20] is not secure too. To the best of our knowledge, there are only three CLGSC schemes in the literature till date.

There are two types of attackers that are generally considered in certificateless cryptography. Type I adversary does not have access to master secret key, however, he may request public keys and replace public keys with values of his choice. Type II adversary does have access to master secret key and can compute partial private key of any user by himself, but may not replace public keys of entities. In 2007, Au et al. [22] introduced a new Type-II adversary named malicious-but-passive key generation center (KGC). This KGC may be malicious at the very beginning of the setup stage of the system and may generate its master public/secret key pair maliciously so that he can launch the Type II attack more easily in the later stage of the system. Fortunately, some cryptographic schemes secure against the malicious-but-passive KGC attacks had been constructed, for example, Hwang et al.'s certificateless encryption scheme [23], and Xiong et al.'s certificateless signature scheme [24]. Weng et al.'s certificateless signcryption scheme [25]. However, it is an interesting thing to construct a CLGSC scheme secure against malicious-but-passive KGC attacks. In the security model of [18–20], they only cover honest-but-curious KGC, as originally defined by Al-Riyami and Paterson [2]. Motivated by this, we introduce a formal security model for CLGSC schemes secure against the malicious-but-passive KGC attacks and propose a novel scheme. Our scheme is proved to be secure assuming GBDH problem, GDH' problem and CDH problem are hard in random oracle model. Performance analysis shows that the proposed scheme is efficient and practical.

The paper is organized as follows. In Sect. 2, the preliminaries are reviewed and the new security model is given. In sect. 3, we propose a concrete scheme under the new security model, and then prove its security under this new model and analyze its performance. We conclude the paper in Sect. 4.

2 Preliminaries

Our scheme relies on bilinear groups and we briefly recall their definition below. We restrict our attention to the symmetric case where $G_1 \cong G_2$ and we may consider a common generator P for them. The following four definitions are quoted from [4].

Definition 1 A bilinear group description Γ is a tuple $(p, G_1, G_2, G_T, e, P_1, P_2)$ where:

- G_1, G_2 and G_T are groups of order p with efficiently computable group laws.
- $e : G_1 \times G_2 \to G_T$ is an efficiently computable non-degenerate bilinear map.
- P_1 and P_2 are generators of G_1 and G_2 respectively.

In practice G_1 and G_2 will be related to the (additive) group of points on an elliptic curve and G_T will be a subgroup of the (multiplicative) group of a finite field. Hence, we use additive notation for G_1 and G_2 and multiplicative notation for G_T.

Definition 2 Given a bilinear group description Γ, we say GBDH assumption holds if the advantage of any probabilistic polynomial time (PPT) adversary is negligible.

$Adv_{\Gamma}^{GBDH}(A, q_{DBDH}) := \Pr[T = e(P, P)^{abc} | a, b, c \leftarrow Z_p; T \leftarrow A^{O_\Gamma}(\Gamma, aP, bP, cP)]$. Here O_Γ denotes a decision bilinear Diffie–Hellman oracle which on input a four-tuple (aP, bP, cP, T) outputs 1 if $T = e(P, P)^{abc}$ and 0 otherwise. By q_{DBDH} we denote the maximum number of queries that A asks its decision oracle.

Definition 3 Given a bilinear group description Γ, we say CDH assumption in the presence of a decision bilinear Diffie–Hellman oracle (GDH') holds in G_1 if the advantage of any PPT adversary is negligible. $Adv_{\Gamma}^{GDH'}(A, q_{DBDH}) := \Pr[Q = abP | a, b \leftarrow Z_p; Q \leftarrow A^{O_\Gamma}(\Gamma, aP, bP)]$, Here O_Γ and q_{DBDH} are as in the above definition.

Definition 4 Given a bilinear group description Γ, we say CDH assumption holds in G_1 if the advantage of any PPT adversary is negligible.

$$Adv_{\Gamma}^{CDH}(A) := \Pr[Q = abP | a, b \leftarrow Z_p; Q \leftarrow A(\Gamma, aP, bP)].$$

2.1 Framework of CLGSC

A CLGSC scheme is defined by the following six PPT algorithms.

- *Setup* (1^k): Given a security parameter k, it generates a master public/secret key pair (m_{pk}, m_{sk}) and global parameters params.
- *Extract-partial-private-key* (*ID, m_{sk}, params*). Given a user identity ID, master secret key m_{sk}, params, it returns a partial private key D.
- *Generate-user-keys* (*ID, params*). Given a user ID, params, it returns a public key of the identity PK, and a secret value x.
- *Set-private-key* (*D, x, params*). Given a partial private key D, a secret value x and params, it returns the full private key S.

- *GSC*. This algorithm has three scenarios: signcryption, signature and encryption.

  Signcryption mode: if user A transmits a message m confidentially and authenticately to B, the input is (S_A, m, ID_B), and outputs $\sigma = GSC(S_A, m, ID_B) = signcrypt(S_A, m, ID_B)$.
  Signature mode: if user A wants to sign a message m without definite receiver, the input is (S_A, m, ID_Φ), where ID_Φ means the receiver is null, the output is $\sigma = GSC(S_A, m, ID_\Phi) = sign(S_A, m)$.
  Encryption mode: if someone wants to send message m to B confidentially, the input is (S_Φ, m, ID_B), where S_Φ means the sender is null. The output is $\sigma = GSC(S_\Phi, m, ID_B) = encrypt(m, ID_B)$.

- *UGSC*. Given σ, if it is valid, the receiver B unsigncrypts (or decrypts) the ciphertext and returns m or true of the signature on m by A, otherwise return $\perp$ means fail.

2.2 Security model of CLGSC

In the security model of [18–20], they only cover honest-but-curious KGC as described before. But in a real environment, the KGC generates the public parameters and the master secret key by itself, while in a security game the simulation algorithm generates the public parameters and the master secret key which are then given to an adversary. So we can see that there is a gap between the real environment and the simulation environment. Therefore, to simulate a malicious-but-passive KGC, we adopt the modification from Au et al. [22] to allow a Type II adversary to generate all the public parameters and the master secret key. There are six oracles which can be accessed by the adversaries as follows.

Partial private key extraction: A submits an identity ID_U, and C returns a partial private key D_U for that identity, generated using the extract-partial-private-key algorithm. Note that A-II does not need this oracle because it has the master secret key and can compute partial private key for any user.
Public key extraction: A submits an identity ID_U, C computes the corresponding public key PK_U and sends it to A. If such a key does not yet exist, it is constructed using the Generate-User-Keys algorithm.
Private key extraction: A submits the identity ID_U, C computes the corresponding private key S_U and sends it to A. If such a key does not yet exist, it is constructed using the appropriate algorithms. Note that if A is Type-I adversary then A is not allowed to extract the full private key of any identity for which corresponding public key has been replaced, because in this case the challenger is not able to provide the full private key of that user.
Public key replacement: For any identity ID_U, A computes the new public key PK'_U by choosing a new secret value x'_U of his choice and replaces PK_U. Note that if A is Type-II adversary then A cannot replace public key of any user.
GSC queries: A submits two identities ID_A, ID_B and a message m. Challenger C runs GSC algorithm and returns the output σ to A.
UGSC queries: A submits two identities ID_A, ID_B along with σ to the challenger C. C runs the UGSC algorithm and returns the output of UGSC to A.

Note it is possible that the public key PK_A (or PK_B) has been replaced earlier by Type-I adversary A in GSC (or UGSC) queries. If so, A has to submit the corresponding secret value to C for the correctness of these oracles.

Confidentiality

The notion of security with respect to confidentiality is indistinguishability of ciphertexts under adaptive chosen ciphertext attacks (IND-CCA2). For CLGSC this notion is captured by the following game played between challenger C and adversary A. We define two games, one for A-I and the other one for A-II.

2.2.1 GAME 1 (IND-CLGSC-CCA2-I)

Initialization: C runs the setup algorithm on input a security parameter k, and gives public parameters params to the adversary A-I. C keeps the master private key secret.
Find stage: The adversary A-I asks the above oracles adaptively.
Challenge: A-I submits two distinct messages m_0 and m_1 of equal length, a sender's identity ID_A^* and a receiver's identity ID_B^* on which he wishes to be challenged. A-I must have made no private key extraction query or partial private key extraction query on ID_B^*, also $ID_B^* \neq ID_\Phi$ for the confidentiality game. C picks randomly a bit $b \in \{0, 1\}$, runs the GSC algorithm with message m_b under ID_A^* and ID_B^* and returns the output σ^* to A-I.
Guess stage: A-I asks queries adaptively again as in the find stage. It is not allowed to extract the private key or partial private key corresponding to ID_B^* and it is not allow to make an UGSC query on σ^* with sender ID_A^* and receiver ID_B^* unless the public key PK_A^* or PK_B^* has been replaced after the challenge.
Eventually, A-I outputs a bit b' and wins the game if $b = b'$. A-I's advantage is defined as $Adv_{A-I}^{IND-CLGSC-CCA2-I} = 2\Pr[b = b'] - 1$.

Note: In confidentiality game, we only need to consider encryption mode and signcryption mode of CLGSC scheme. In the above challenge phase, the sender ID_A^* can be vacant. In this case, algorithm runs in encryption mode otherwise it runs in signcryption mode, so encryption mode and signcryption mode share the same game.

Definition 5 A CLGSC scheme is said to IND-CLGSC-CCA2-I secure in encryption or signcryption mode if for all PPT adversary A-I, it is negligible to win the game.

2.2.2 GAME 2 (IND-CLGSC-CCA2-II)

Let C be the game simulator and k be a security parameter.

1. C executes A-II on 1^k and a special tag *master-key-gen*. A-II generates a master key pair (M_{SK}, M_{PK}), sets params = (Γ, M_{PK}) and provides (Γ, M_{PK}) and M_{SK} to C. Note that A-II is not allowed to query any oracle in this phase.
2. C invokes A-II again with 1^k but with another tag *choose*. During the simulation, A-II can make the above oracles adaptively. At the end of this phase, A-II outputs two equal-length messages (m_0, m_1) and a sender's identity ID_A^* and a receiver's identity ID_B^* on which he wishes to be challenged. A-II must have made no private key extraction query on ID_B^*, also $ID_B^* \neq ID_\Phi$ for the confidentiality game.
3. C picks randomly a bit $b \in \{0, 1\}$ and runs A-II on input a challenge ciphertext σ^* and a tag *guess* where $\sigma^* \leftarrow GSC(m_b, ID_A^*, ID_B^*)$.
4. A-II asks queries adaptively again as in step 2. It is not allowed to extract the private key corresponding to ID_B^* and it is not allow to make an UGSC query on σ^* with sender ID_A^* and receiver ID_B^*. Eventually, A-II outputs a bit b' and wins the game if $b = b'$. A-II's advantage is defined as $Adv_{A-II}^{IND-CLGSC-CCA2-II} = 2\Pr[b = b'] - 1$.

Note: In step 2, the sender ID_A^* can be vacant. In this case, algorithm runs in encryption mode otherwise it runs in signcryption mode, so encryption mode and signcryption mode share the same game.

Definition 6 A CLGSC scheme is said to IND-CLGSC-CCA2-II secure in encryption or signcryption mode if for all PPT adversary A-II, it is negligible to win the game.

Unforgeability

The notion of security with respect to authenticity is existential unforgeability against chosen message attacks (EUF-CMA). For CLGSC this notion is captured by the following game played between challenger C and adversary A.

2.2.3 GAME 3 (EUF-CLGSC-CMA-I)

Initialization: Same as in GAME 1.
Queries: The adversary A-I asks a polynomially bounded number of the above oracles adaptively.
Forgery: Finally, A-I produces a triplet (ID_A, ID_B, σ) that was not obtained from GSC query during the game and for which private key or partial private key of ID_A was not exposed, also $ID_A \neq ID_\Phi$ for unforgeability game. A-I wins the game if the result of $UGSC(\sigma, ID_B, S_B, PK_B, ID_A, PK_A)$ is not the $\perp$ symbol. A's advantage is its probability of victory.

Note: In unforgeability game, we only need to consider signature mode and signcryption mode of CLGSC scheme. In the above forgery phase, the sender ID_B^* can be vacant. In this case, algorithm runs in signature mode otherwise it runs in signcryption mode, so signature mode and signcryption mode share the same game.

Definition 7 A CLGSC scheme is said to EUF-CLGSC-CMA-I secure in signature or signcryption mode if for all PPT adversary A-I, it is negligible to win the game.

2.2.4 GAME 4 (EUF-CLGSC-CMA-II)

Step 1 is the same as in GAME 2.

2: C invokes A-II again with 1^k but with another tag *forge*. A-II can ask a polynomially bounded number of the above oracles adaptively. Finally, A-II produces a triplet (ID_A, ID_B, σ) that was not obtained from GSC query during the game and for which private key of ID_A was not exposed, also $ID_A \neq ID_\Phi$ for unforgeability game.

A-II wins the game if the result of $UGSC(\sigma, ID_B, S_B, PK_B, ID_A, PK_A)$ is not the $\perp$ symbol. A-II's advantage is its probability of victory.

Note: In step 2, the sender ID_B^* can be vacant. In this case, algorithm runs in signature mode otherwise it runs in signcryption mode, so signature mode and signcryption mode share the same game.

Definition 8 A CLGSC scheme is said to EUF-CLGSC-CMA-II secure in signature or signcryption mode if for all PPT adversary A-II, it is negligible to win the game.

3 A new CLGSC scheme (N-CLGSC)

Our scheme is also referenced by scheme [4] as scheme [20]. [21] points out scheme [4] is not secure, so we made some modifications to overcome the security flaw.

3.1 Scheme

Setup (1^k): given a security parameter 1^k, the KGC chooses two groups G_1 and G_2 of prime order q, a generator $\mathbf{P}$ of G_1, a bilinear map $e : G_1 \times G_1 \to G_2$, 4 hash functions as $H_1 : \{0,1\}^* \to G_1$, $H_2 : \{0,1\}^* \to \{0,1\}^k$, $H_3 : \{0,1\}^* \to G_1$, $H_4 : \{0,1\}^* \to G_1$, where k denotes the number of bits to represent a message. KGC chooses random $s \in Z_q^*$ as master secret key and set $P_{Pub} = sP$. Define a special function $f(ID)$,when $ID \in \phi$, $f(ID) = 0$ else $f(ID) = 1$ KGC publishes the system parameters as $\{G_1, G_2, e, q, f, P, P_{Pub}, H_1, H_2, H_3, H_4\}$. Note that KGC may be malicious at this stage.
Extract-partial-private-key: given ID_i, the partial private key of the user with identity ID_i is computed by KGC as $D_i = sQ_i = sH_1(ID_i)$.
Generate-user-keys: given D_i, the user with identity ID_i chooses random $x_i \in z_q^*$ and computes his public key $PK_i = x_i P$.
Set-private-key: user sets his private key $SK_i =< x_i, D_i >$.
GSC (m, ID_A, ID_B):

1 computes $f(ID_A)$, $f(ID_B)$, chooses random $r \in Z_q^*$, computes $U = rP, w = e(P_{Pub}, Q_B)^{rf(ID_B)}$;
2 computes $h = f(ID_B)H_2(U, w, rPK_B, ID_A, PK_A, ID_B, PK_B)$, $V = m \oplus h$;
3 computes $H = H_3(U, V, ID_A, PK_A, ID_B, PK_B)$, $H' = H_4(U, V, ID_A, PK_A, ID_B, PK_B)$;
4 computes $W = f(ID_A)D_A + rH + f(ID_A)x_A H'$, and returns ciphertext $c = (U, V, W)$.

UGSC (U, V, W, ID_A, ID_B):

1 computes $f(ID_A)$, $f(ID_B)$, $H = H_3(U, V, ID_A, PK_A, ID_B, PK_B)$, $H' = H_4(U, V, ID_A, PK_A, ID_B, PK_B)$;
2 if $e(P, W) \neq e(P_{Pub}, Q_A)^{f(ID_A)} e(U, H) e(PK_A, H')^{f(ID_A)}$ returns $\perp$ else
3 computes $w = e(U, D_B)^{f(ID_B)}$, $h = f(ID_B)H_2(U, w, x_B U, ID_A, PK_A, ID_B, PK_B)$, $m = V \oplus h$, and returns m.

3.2 Adaptation

CLGSC is an adaptive scheme that can seamlessly switch to different modes according to the inputs of users' identities, applications need not care about all of these works. Note that the scheme seamlessly switches to three modes without any other additional operation.

Signcryption mode: when $ID_A \notin \phi$, $ID_B \notin \phi$ then $f(ID_A) = 1$, $f(ID_B) = 1$, algorithm runs in signcryption mode.
Encryption mode: when $ID_A \in \phi$, $ID_B \notin \phi$ then $f(ID_A) = 0$, $f(ID_B) = 1$, algorithm runs in encryption mode. In this case, $W = rH$, the check equation becomes: $\hat{e}(P, W) = \hat{e}(U, H)$.
Signature mode: when $ID_A \notin \phi$, $ID_B \in \phi$ then $f(ID_A) = 1$, $f(ID_B) = 0$, algorithm runs in signature mode. In this case, $V = m \oplus h = m \oplus f(ID_B)H_2(U, w, rPK_B, ID_A, PK_A, ID_B, PK_B) = m \oplus 0 = m$, the ciphertext is $c = (U, V, W) = (U, m, W)$, namely (U, W) is the signature of m.

3.3 Proof of Confidentiality of N-CLGSC

Theorem 1 *In the random oracle model, if a PPT attacker A-I has non-negligible advantage in winning the IND-CLGSC-CCA2-I game against the scheme proposed above in encryption mode or signcryption mode, then there exists an algorithm B which uses A-I to solve the GBDH problem such that:* $Adv_{CLGSC}^{IND-CCA2-I}(A-I) \leq q_T Adv_{\Gamma}^{GBDH}(B, q_D^2 + 2q_D q_2 + q_2) + q_{SC}(q_{SC} + q_D + q_3 + 1)/2^k$, *where* $q_T = q_1 + q_X + q_{SK} + 2q_D + 2q_{SC} + 2$. *Here* $q_1, q_2, q_3, q_X, q_{SK}, q_{SC}$ *and* q_D *are the maximum number of queries that the adversary could place to* H_1, H_2, H_3, *partial private key extraction, private key extraction, GSC and UGSC oracles.*

Proof See the Appendix A. □

Theorem 2 *In the random oracle model, if a PPT attacker A-II has non-negligible advantage in winning the IND-CLGSC-CCA2-II game against the scheme proposed above in encryption mode or signcryption mode, then there exists an algorithm B which uses A-II to solve the CDH problem such that:* $Adv_{CLGSC}^{IND-CCA2-II}(A-II) \leq q_T Adv_{\Gamma}^{CDH}(B) + q_{SC}(q_{SC} + q_D + q_3 + 1)/2^k$, *where* $q_T = q_{PK} + q_{SK} + 2q_D + 2q_{SC} + 2$. *Here* q_{PK} *is the maximum number of queries that the adversary could place to request public key oracle and others are as before.*

Proof See the Appendix B. □

3.4 Proof of Unforgeability of N-CLGSC

Theorem 3 *In the random oracle model, if a PPT attacker A-I has non-negligible advantage in winning the* **EUF-CLGSC-CMA-***I game against the scheme proposed above in signature mode or signcryption mode, then there exists an algorithm B which uses A-I to solve the* GDH' *problem such that:* $Adv_{CLGSC}^{EUF-CMA-I}(A-I) \leq q_T Adv_{\Gamma}^{GDH'}(B, q_D^2 + 2q_D q_2) + (q_{SC}(q_{SC} + q_D + q_3 + 1) + 2)/2^k$, *where* $q_T = q_1 + q_X + q_{SK} + 2q_D + 2q_{SC} + 1$ *and various q's are as before.*

Proof See the Appendix C. □

Theorem 4 *In the random oracle model, if a PPT attacker A-II has non-negligible advantage in winning the* **EUF-CLGSC-CMA-***II game against the scheme proposed above in signature mode or signcryption mode, then there exists an algorithm B which uses A-II to solve the CDH problem such that:* $Adv_{CLGSC}^{EUF-CMA-II}(A-II) \leq q_T Adv_{\Gamma}^{CDH}(B) + (q_{SC}(q_{SC} + q_D + q_3 + 1) + 2)/2^k$, *where*
$q_T = q_{PK} + q_{SK} + 2q_D + 2q_{SC} + 1$ *and various q's are as before.*

Proof See the Appendix D. □

3.5 Efficiency of N-CLGSC

Since computation time and ciphertext size are two important factors affecting the efficiency, we present the comparison with respect to them. Our scheme can be seen as the improvement of scheme [20], so there are only three CLGSC schemes in the literature till date. Table 1

Table 1 Efficiency comparison with other CLGSC schemes

Scheme	Ciphertext size	GSC			UGSC		
		E	M	P	E	M	P
Scheme [18]	$2\|G_1\| + \|m\| + \|ID\| + \|G_2\| + \|p\|$	3	2	0	1	1	2
Scheme [19]	$2\|G_1\| + \|m\| + \|ID\| + \|G_2\|$	2	3	0	1	3	2
Ours	$2\|G1\| + \|m\|$	1	4	0(+1)	0	1	4(+1)

shows the comparison. In Table 1, M denotes the number of point multiplications in G_1, E denotes the number of exponentiations in G_2, P denotes the number of pairing computations, (+1) denotes pre-computation of pairing, and $|G_1|$, $|G_2|$, |m|, |ID|, |p| denote the size of an element in G_1, the size of an element in G_2, the length of message m, the length of identity ID and the size of an element in Z_P^*. From Table 1, it shows that our scheme has the shortest ciphertext size and is of high efficiency too.

4 Conclusion

In this paper, we extend the security model of CLGSC and propose a concrete scheme, and then prove its security in the random oracle model under the GBDH and CDH assumptions. According to the comparison with other CLGSC schemes, the new scheme is efficient and practical.

Acknowledgments This work is supported by the National Nature Science Foundation of China under Grant No. 61103231.

Appendix

A Proof of Theorem 1

Proof On receiving the GBDH challenge tuple (Γ, aP, bP, cP), where the generator is P, B sets $M_{PK} = aP$ and $params = (\Gamma, M_{PK})$ and passes them on to A-I. B chooses an index ℓ uniformly at random in $\{1, \ldots, q_T\}$, and answers various oracle queries as follows:

H_1 queries: On the i-th non-repeat query ID, if $i \neq \ell$ B chooses $r \in Z_p$ uniformly at random and sets $Q_{ID} = rP$. It then adds (i, ID, r) to a list L_1 which is initially empty and returns Q_{ID}. Otherwise, it returns $Q_{ID_\ell} = bP$ and adds $(\ell, ID, \perp)$ to L_1.
Extract partial secret key queries: For each new query ID, B calls H_1 on ID and obtains (i, ID, r). If $i = \ell$ then B aborts the simulation; Otherwise, B returns $D = raP$.
Request public key queries: For each query ID, B checks in list L_K, which is initially empty, if there is a tuple (ID, PK, x). If so, B returns PK. Otherwise, B generates a new key pair, updates the list L_K, and returns the public key.
Replace public key queries: On input (ID, PK), B updates L_K with tuple (ID, PK, $\perp$).
Extract private key queries: For each new query ID, B calls H_1 on ID and obtains (i, ID, r). If $i = \ell$ then B aborts the simulation; Otherwise, B searches L_K for the entry (ID, PK, x), generating a new key pair if this does not exist, and returns (x, raP).

H_3 *queries*: For each new query $(U, V, ID_A, PK_A, ID_B, PK_B)$, B generates a random value $t \in Z_p$, updates an initially empty list L_3 with the input, t and tP and returns tP.
H_4 *queries*: For each new query $(U, V, ID_A, PK_A, ID_B, PK_B)$, B generates a random value $s \in Z_p$, updates an initially empty list L_4 with the input, s and sP and returns sP.
H_2 *queries*: For each new query $(U, T, R, ID_A, PK_A, ID_B, PK_B)$, B proceeds as follows:

1. It checks if the decision bilinear Diffie–Hellman oracle returns 1 when queried with the tuple (aP, bP, cP, T). If this is the case, B returns T and stops.
2. B goes through the list L_2 with entries $(U, *, R, ID_A, PK_A, ID_B, PK_B, h)$, for different values of h, such that the decision bilinear Diffie–Hellman oracle returns 1 when queried on the tuple (aP, bP, U, T). Note that in this case $ID_B = ID_\ell$. If such a tuple exists, it returns h (and replaces the symbol * with T).
3. It goes through L_2 with entries $(U, T, *, ID_A, PK_A, ID_B, PK_B, h)$, for different values of h, such that $e(U, PK_B) = e(P, R)$. If such a tuple exists, it returns h (and replaces the symbol * with R).
4. If B reaches this point of execution, it returns a random h and updates the list L_2, which is initially empty, with a tuple containing the input and return values.

GSC queries: For each new query (m, ID, ID'), if $ID \in \phi$, $ID' \notin \phi$, it equals to encryption oracle, which just need public parameters; else if $ID \notin \phi$, consider two cases:

Case 1: $ID \neq ID_\ell$, B simply produces the GSC ciphertext as normal because B can get the private key of ID.
Case 2: $ID = ID_\ell$

1. B generates two random values $u, v \in Z_P$, sets $U = vaP$, calculates $T = e(U, r'M_{PK})$, obtaining (j, ID', r') by calling H_1 on ID'.
2. It goes through list L_2 looking for an entry $(U, T, R, ID_\ell, PK_\ell, ID', PK', h)$ for some R such that $e(U, PK') = e(P, R)$. If such an entry exists, it calculates $V = m \oplus f(ID')h$. Otherwise it uses a random h and updates the list L_2 with $(U, T, *, ID_\ell, PK_\ell, ID', PK', h)$.
3. Then B defines the hash value $H_3(U, V, ID_\ell, PK_\ell, ID', PK')$ as $H = v^{-1}(uP - bP)$, aborting the simulation if such a hash queries has been responded with a different value before. This means that B updates list L_3 with tuple $(U, V, ID_\ell, PK_\ell, ID', PK', \perp, H)$. Finally, B sets $W = uaP + sPK$, where s is the value obtained by querying H_4 on $(U, V, ID_\ell, PK_\ell, ID', PK')$ and returns (U,V,W). Note that this is a valid GSC ciphertext.

UGSC queries: For each new query (U, V, W, ID, ID'), it executes the verification part of the UGSC algorithm, which just needs public parameters. It returns $\perp$ and stops if verification does not succeed; else if $ID' \in \phi$, $ID \notin \phi$, it equals to signature verification oracle, which just need public parameters; else if $ID' \notin \phi$, consider two cases:

Case 1: $ID' \neq ID_\ell$, B just UGSC as normal because B can get the private key of ID';
Case 2: $ID' = ID_\ell$

1. It calculates $R = x'U$, obtaining x' (and hence PK') from either the adversary or by calling the request public key oracle.
2. Because B cannot get the partial private key of ID', in order to return a consistent answer, B goes through L_2 and looks for a tuple $(U, T, R, ID, PK, ID_\ell, PK', h)$, for different values of T, such that the decision bilinear Diffie–Hellman oracle returns

1 when queried on (aP, bP, U, T). If such an entry exists, the correct pairing value is found and B decrypts using the hash h.

3. If B reaches this point of execution, it places the entry $(U, *, R, ID, PK, ID_\ell, PK', h)$ for a random h on list L_2 and decrypts using this h. The symbol * denotes an unknown value of pairing. Note that the identity component of all entries with a * is ID_ℓ.

Eventually, A-I outputs two messages (m_0, m_1) and two identities ID_S^* and ID_R^*. B places a query on H_1 with input ID_R^*. If the index of ID_R^* is not ℓ, B fails; Otherwise it proceeds to construct a challenge as follows. It obtains from L_K the public key PK corresponding to ID_S^*. Then it sets $U^* = cP$, selects a random bit b and a random hash h^* and sets $V^* = m_b \oplus h^*$. The component W^* is set to be $D_S + rH + x_S H' = D_S + tcP + sPK$ where t is obtained from L_3, s is obtained from L_4 and D_S is calculated by calling the partial secret key extraction oracle on ID_S^*.
In the second stage, A-I's queries are answered as before. Eventually, A-I will output its guess. Since ℓ is independent of adversary's view, and the list L_1 can be easily seen to have at most q_T elements, with probability $1/q_T$ the adversary will output an identity ID_ℓ with index ℓ. If this event occurs, the simulation is perfect unless the adversary queries H_2 on the challenge-related tuple $(U^*, T^*, R^*, ID_S^*, PK_S^*, ID_\ell, PK_\ell)$. Since the hash function H_2 is modeled as a random oracle, the adversary will not have any advantage if this tuple does not appear on L_2. However, if this happens, B will win the game due to the first step in the simulation of H_2. The theorem follows from this observation and the fact that the total number of decision bilinear Diffie–Hellman oracle calls that B makes is at most $q_D^2 + 2q_D q_2 + q_2$. If B's simulation triggers a collision in its GSC simulation of H_3, since the maximum size of L_3 is $q_{SC} + q_D + q_3 + 1$, we can upper bound the probability this occurs as $q_{SC}(q_{SC} + q_D + q_3 + 1)/2^k$.

Note: In the above challenge phase, the sender ID_S^* can be vacant. In this case, algorithm runs in encryption mode otherwise it runs in signcryption mode, so the proof is suitable for the two modes. □

B Proof of Theorem 2

Proof On receiving the CDH challenge tuple (Γ, aP, bP) with generator P, in the first step, A-II is executed and A-II generates a master key pair (M_{SK}, M_{PK}), sets params= (Γ, M_{PK}) and provides (Γ, M_{PK}) and M_{SK} to B. B chooses an index ℓ uniformly at random in $\{1, \ldots, q_T\}$ and answers various oracle queries made by A-II as follows:

H_1 queries: On the non-repeat query ID, B chooses $r \in Z_P$ uniformly at random and sets $Q_{ID} = rP$. It then adds (ID,r) to a list L_1 which is initially empty and returns Q_{ID}.
Request public key queries: On the i-th non-repeat query ID, if $i \neq \ell$, B generates a new key pair (x, PK), updates the list L_K with (i, ID, x, PK). If $i = \ell$ B returns aP and adds $(\ell, ID, aP, \perp)$ to L_K.
Extract private key queries: For each new query ID, B calls request public key on ID obtaining (i, ID, PK, x). If $i = \ell$, B aborts the simulation; Otherwise, it calls H_1 on ID and gets (ID, r). It returns $(x, rM_{SK}P)$.
H_3, H_4 queries are the same as Theorem 1.
H_2 **queries**: For each new query $(U, T, R, ID_A, PK_A, ID_B, PK_B)$, B proceeds as follows:

1. It checks if $e(aP, bP) = e(P, R)$. If so, B returns R and stops.
2. B goes through list L_2 looking for entries $(U, T, *, ID_A, PK_A, ID_B, PK_B, h)$, such that $e(U, bP) = e(P, R)$. Note that in this case $ID_B = ID_\ell$. If such a tuple exists, it returns h (and replaces the symbol * with R).
3. If B reaches this point of execution, it returns a random h and updates the list L_2, which is initially empty, with a tuple containing the input and return values.

GSC queries: For each new query (m, ID, ID'), if $ID \in \phi$, $ID' \notin \phi$, it equals to encryption oracle, which just need public parameters; else if $ID \notin \phi$, consider two cases:

Case 1: $ID \neq ID_\ell$, B simply produces the GSC ciphertext as normal because B can get the private key of ID.
Case 2: $ID = ID_\ell$

1. B generates two random values $u, v \in Z_P$, sets $U = vaP$, calculates $T = e(U, M_{SK} Q_{ID'})$.
2. It goes through list L_2 looking for an entry $(U, T, R, ID_\ell, PK_\ell, ID', PK', h)$ for some R such that $e(U, PK') = e(P, R)$. If such an entry exists, it calculates $V = m \oplus f(ID')h$. Otherwise it uses a random h and updates the list L_2 with$(U, T, *, ID_\ell, PK_\ell, ID', PK', h)$.
3. Then B defines the hash value $H_3(U, V, ID_\ell, PK_\ell, ID', PK')$ as $H = v^{-1}(uP - H_4)$, aborting the simulation if such a hash response has been given before. This means that B updates list L_3 with tuple $(U, V, ID_\ell, PK_\ell, ID', PK', \perp, H)$. Finally, B sets $W = D_S + uaP$ and returns (U, V, W). Note that this is a valid GSC ciphertext.

UGSC queries: For each new query (U, V, W, ID, ID'), it executes the verification part of the UGSC algorithm, which just needs public parameters. It returns $\perp$ if verification does not succeed; else if $ID' \in \phi$, $ID \notin \phi$, it equals to signature verification oracle, which just need public parameters; else if $ID' \notin \phi$, consider two cases:

Case 1: $ID' \neq ID_\ell$, B just UGSC as normal because B can get the private key of ID';
Case 2: $ID' = ID_\ell$

1. It calculates $T = e(U, r'M_{PK})$, where (ID', r') is obtained from H_1.
2. Because B cannot compute the correct value of R. in order to return a consistent answer, B goes through L_2 and looks for a tuple $(U, T, R, ID, PK, ID_\ell, PK', h)$, for different values of R, such that $e(U, bP) = e(P, R)$. If such an entry exists, the correct value of R is found and B decrypts using the hash value h.
3. If B reaches this point of execution, B places the entry $(U, T, *, ID, PK, ID_\ell, PK', h)$ for a random h on list L_2 and decrypts using this h.

Eventually, A-II outputs two messages (m_0, m_1) and two identities ID_S^* and ID_R^*. B queries the request public key oracle on ID_R^* and receives (j, ID_R^*, PK^*, x^*). If $j \neq \ell$, it fails; otherwise it proceeds to construct a challenge as follows. It obtains the public key PK for ID_S^* by calling the request public key oracle. It sets $U^* = bP$, selects a random bit b and a random hash value h^* and sets $V^* = m_b \oplus h^*$. The component W^* is set to be $D_S + rH + x_S H' = D_S + tbP + sPK$, where D_S is obtained by calling the extract partial secret key oracle, t is obtained from L_3 and s is obtained from L_4.
In the second stage, A-II's queries are answered as before. Eventually, A-II will output its guess. Since ℓ is independent of adversary's view, with probability $1/q_T$ the adversary will output an identity ID_R^* with index ℓ. If this event occurs, the simulation is perfect unless the challenge-related tuple $(U^*, T^*, R^*, ID_S^*, PK_S^*, ID_\ell, PK_\ell)$ is queried from

H_2. However, since the hash function H_2 is modeled as a random oracle, the adversary will not have any advantage if this entry does not appear on L_2. However, if this happens, B will win the game due to its simulation of H_2. The theorem follows from this observation and the fact that the maximum length of the list L_K is q_T. If B's simulation triggers a collision in its GSC simulation of H_3, since the maximum size of L_3 is $q_{SC}+q_D+q_3+1$, we can upper bound the probability this occurs as $q_{SC}(q_{SC}+q_D+q_3+1)/2^k$.

Note: In the above challenge phase, the sender ID_S^* can be vacant. In this case, algorithm runs in encryption mode otherwise it runs in signcryption mode, so the proof is suitable for the two modes. □

C Proof of Theorem 3

Proof To prove this theorem, we construct an algorithm B which uses A-I to solve the GDH' problem over G_1. B receives a GDH' problem instance (Γ, aP, bP), with generator P, it sets $M_{SK} = aP$ and provides params = (Γ, M_{PK}) to A-I. B then chooses an index ℓ uniformly at random in $\{1, \ldots, q_T\}$, and answers various oracle queries as follows:
H_1, H_3, H_4, *Extract partial secret key, request public key, replace public key, extract private key, GSC, UGSC queries* are the same as Theorem 1.
H_2 *queries*: For each new query $(U, T, R, ID_A, PK_A, ID_B, PK_B)$, B proceeds as follows:

1. It checks if $e(aP, bP) = e(P, R)$. If so, B returns R and stops.
2. B goes through the list L_2 with entries $(U, *, R, ID_A, PK_A, ID_B, PK_B, h)$, for different values of h, such that the decision bilinear Diffie–Hellman oracle returns 1 when queried on the tuple (aP, bP, U, T). Note that in this case $ID_B = ID_\ell$. If such a tuple exists, it returns h (and replaces the symbol * with T).
3. It goes through list L_2 with entries entry $(U, T, *, ID_A, PK_A, ID_B, PK_B, h)$, for different values of h, such that $e(U, PK_B) = e(P, R)$. If such a tuple exists, it returns h (and replaces the symbol * with R).
4. If B reaches this point of execution, it returns a random h and updates the list L_2, which is initially empty, with a tuple containing the input and return values.

Eventually, A-I outputs a signcryption (U^*, V^*, W^*) from sender ID_S^* to receiver ID_R^*. B now calls H_1 on ID_S^* and checks if $ID_S^* = ID_\ell$ and if this is not the case it aborts; otherwise, it obtains PK_S^* by calling the request public key oracle on ID_S^* and retrieves t^* and s^* from lists L_3 and L_4 by querying H_3 and H_4 on $(U^*, V^*, ID_\ell, PK_S^*, ID_R^*, PK_R^*)$. Note that if A-I succeeded, then the verification condition holds:

$$\begin{aligned} e(P, W^*) &= e(M_{PK}, Q_{ID_\ell})e(U^*, H^*)e(PK_S^*, H'^*) \\ e(P, W^*) &= e(aP, bP)e(U^*, t^*P)e(PK_S^*, s^*P) \\ e(P, abP) &= e(P, W^* - t^*U^* - s^*PK_S^*) \end{aligned}$$

Thus B can recover $abP = W^* - t^*U^* - s^*PK_S^*$.

Let us now analyze the probability that B succeeds in solving the GDH' problem instance. For this to happen, the simulation must run until the end of the game, the adversary must pick a specific identity as ID_S^*, and it must query the hash functions H_3 and H_4 to properly construct the forgery. The probability that A-I is able to produce a forgery without querying both hash functions is upper bounded by $2/2^k$.

The probability that B aborts the simulation is related with the following events:

– A-I places a partial key extraction on ID_ℓ.
– A-I places a full secret key extraction on ID_ℓ.
– B wants to simulate a GSC query and this leads to an inconsistency in the H_3 simulation.

Note that if A-I places either of the first two fatal queries, then it could not possibly use ID_ℓ as the sender identity in the forgery it produces at the end of the game, so we can pinpoint the probability that B does not abort the simulation due to these events and A-I picks the only useful case for solving GDH' as $1/q_T$.

The latter fatal event occurs if B's simulation triggers a collision in its simulation of H_3. Since the maximum size of L_3 is $q_{SC}+q_D+q_3+1$, we can upper bound the probability this occurs as $q_{SC}(q_{SC}+q_D+q_3+1)/2^k$. The result follows by noting that B makes at most $q_D^2+2q_Dq_2$ queries to its decision bilinear Diffie–Hellman oracle.

Note: In the above forgery phase, the sender ID_R^* can be vacant. In this case, algorithm runs in signature mode otherwise it runs in signcryption mode, so the proof is suitable for the two modes. □

D Proof of Theorem 4

Proof To prove this theorem, we construct an algorithm B which uses A-II to solve the CDH problem over G_1. B receives a CDH problem instance (Γ, aP, bP) with generator P, in the first step, A-II is executed and A-II generates a master key pair (M_{SK}, M_{PK}), sets params = (Γ, M_{PK}) and provides (Γ, M_{PK}) and M_{SK} to B. B then chooses an index ℓ uniformly at random in $\{1, \ldots, q_T\}$, and answers various oracle queries as follows:
H_1, H_2, H_3, *Request public key, extract private key, GSC, UGSC queries* are the same as Theorem 2.
H_4 *queries*: For each new query $(U, V, ID_A, PK_A, ID_B, PK_B)$, B generates a random value $s \in Z_P$, updates an initially empty list L_4 with the input, s and sbP and returns sbP.

Eventually, A-II outputs a valid signcryption (U^*, V^*, W^*) from sender ID_S^* to receiver ID_R^*. B now checks if $ID_S^* = ID_\ell$. If this is not the case it aborts; otherwise, it retrieves t^* and s^* from lists L_3 and L_4 by querying H_3 and H_4 on $(U^*, V^*, ID_\ell, PK_S^*, ID_R^*, PK_R^*)$. Note that if A-II succeeded, then the verification condition holds:

$$\begin{aligned} e(P, W^*) &= e(M_{PK}, Q_{ID_\ell})e(U^*, H^*)e(PK_S^*, H'^*) \\ e(P, W^*) &= e(P, D_{ID_\ell})e(U^*, t^*P)e(aP, s^*bP) \\ e(P, s^*abP) &= e(P, W^* - D_{ID_\ell} - t^*U^*) \end{aligned}$$

Thus B can recover $abP = (W^* - D_{ID_\ell} - t^*U^*)/s^*$.

Let us now analyze the probability that B succeeds in solving CDH. For this to happen, the simulation must run until the end of the game, the adversary must pick a specific identity as ID_S^*, and it must query the hash functions H_3 and H_4 to properly construct the forgery. The probability that A-II is able to produce a forgery without querying both hash functions is upper bounded by $2/2^k$. The probability that B aborts the simulation is related with the following events:

– A-II places a full secret key extraction on ID_ℓ.
– B wants to simulate a GSC query and this leads to an inconsistency in the H_3 simulation.

Note that if A-II places the first fatal query, then it could not possible use ID_ℓ as the sender identity in the forgery it produces at the end of the game, so we can pinpoint the probability

that B does not abort the simulation due to this event and A-II picks the only useful case for solving CDH as $1/q_T$.

The latter fatal event occurs if B's simulation triggers a collision in its GSC simulation of H_3. Since the maximum size of L_3 is $q_{SC}+q_D+q_3+1$, we can upper bound the probability this occurs as $q_{SC}(q_{SC}+q_D+q_3+1)/2^k$. The result follows.

Note: In the above forgery phase, the sender ID_R^* can be vacant. In this case, algorithm runs in signature mode otherwise it runs in signcryption mode, so the proof is suitable for the two modes. □

References

1. Shamir A.: Identity-based cryptosystems and signature schemes. In: CRYPTO'84. Lecture Notes in Computer Science, vol. 196, pp. 47–53. Springer, Heidelberg (1984).
2. Al-Riyami S.S., Paterson K.G.: Certificateless public key cryptography. In: Proceedings of ASIACRYPT 2003. Lecture Notes in Computer Science, vol. 2894, pp. 452–473. Springer, Heidelberg (2003).
3. Zheng Y.L.: Digital signcryption or how to achieve cost (Signature & Encryption) Cost (Signature) + Cost (Encryption). In: CRYPTO'97. Lecture Notes in Computer Science, vol. 1294, pp. 165–179. Springer, Heidelberg (1997).
4. Barbosa M., Farshim P.: Certificateless signcryption. In: Proceedings of ASIACCS'2008, pp. 369–372. ACM, New York (2008).
5. Han Y.L., Yang X.Y., Wei P., et al.: ECGSC: elliptic curve based generalized signcryption. In: The 3rd International Conference on Ubiquitous Intelligence and Computing (UIC-2006). Lecture Notes in Computer Science, vol. 4159, pp. 956–965. Springer, Heidelberg (2006).
6. Han Y.L.: Generalization of signcryption for resources-constrained environments. Wirel. Commun. Mobile Comput. **7**(7), 919–931 (2007).
7. ANSI X9.62: Public key cryptography for the financial services industry: the Elliptic Curve Digital Signature Algorithm (ECDSA). (1999).
8. Wang X.A., Yang X.Y., Han Y.L.: Provable secure generalized signcryption. Cryptology ePrint Archive, Report 2007/173. http://eprint.iacr.org (2007).
9. Lai S., Kushwah P.: ID-based generalized signcryption. Cryptology ePrint Archive, Report 2008/084. http://eprint.iacr.org (2008).
10. Yu G., Ma X.X., Shen Y., et al.: Provable secure identity based generalized signcryption scheme. Theor. Comput. Sci. **411**(40–42), 3614–3624 (2010).
11. Kushwah P., Lai S.: An efficient identity based generalized signcryption scheme. Theor. Comput. Sci. **412**(45), 6382–6389 (2011).
12. Yang X.Y., Li M.T., Wei L.X., et al.: New ECDSA-verifiable multi-receiver generalization signcryption. In: The 10th IEEE International Conference on High Performance Computing and Communications, Dalian, pp. 1042–1047 (2008).
13. Han Y.L., Gui X.L.: Adaptive secure multicast in wireless networks. Int. J. Commun. Syst. **22**(9), 1213–1239 (2009).
14. Han Y.L., Gui X.L.: BPGSC: Bilinear pairing based generalized signcryption scheme. In: 2009 Eighth International Conference on Grid and Cooperative Computing, Lanzhou, pp. 76–82 (2009).
15. Zhang C.R., Zhang Y.Q.: Secure and efficient generalized signcryption scheme based on a short ECDSA. In: The Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2010), Darmstadt, pp. 466–469 (2010).
16. Ji H.F., Han W.B., Zhao L.: Identity-based generalized signcryption in standard model. Appl. Res. Comput. **27**(10), 3851–3854 (2010) (in chinese).
17. Ji H.F., Han W.B., Liu L.D.: Identity based generalized signcryption scheme for multiple PKGs in standard model. J. Electron. Inf. Technol. **33**(5), 1204–1210 (2011) (in chinese).
18. Ji H.F., Han W.B., Zhao L.: Certificateless generalized signcryption. Cryptology ePrint Archive, Report 2010/204. http://eprint.iacr.org (2010).
19. Kushwah P., Lai S.: Efficient generalized signcryption schemes. Cryptology ePrint Archive, Report 2010/346. http://eprint.iacr.org (2010).
20. Ji H.F., Han W.B., Zhao L.: Certificateless generalized signcryption. In: Proceedings of 2010 International Colloquium on Computing, Communication, Control, and Management (CCCM2010), vol. 2, Yangzhou (2010).

21. Selvi S.S.D., Vivek S.S., Rangan C.P.: Cryptanalysis of certificateless signcryption schemes and an efficient construction without pairing. Cryptology ePrint Archive, Report 2009/298. http://eprint.iacr.org (2009).
22. Au M.H., Chen J.K., Liu J.K., et al.: Malicious KGC attacks in certificateless cryptography. In: Proceedings of ASIACCS 2007, pp. 302–311. ACM, New York (2007).
23. Hwang Y.H., Liu J.K., Chow S.S.M.: Certificateless public key encryption secure against malicious KGC attacks in the standard model. J. Univers. Comput. Sci. **14**(3), 463–480 (2008).
24. Xiong H., Qin Z.G., Li F.G.: An improved certificateless signature scheme secure in the standard model. Fundam. Inf. **88**, 193–206 (2008).
25. Weng J., Yao G.X., Deng R.H., et al.: Cryptanalysis of a certificateless signcryption scheme in the standard model. Inf. Sci. **181**, 661–667 (2011).