Hindawi
Security and Communication Networks
Volume 2017, Article ID 8405879, 17 pages
https://doi.org/10.1155/2017/8405879

Research Article

Certificateless Key-Insulated Generalized Signcryption Scheme without Bilinear Pairings

Caixue Zhou,[1] Zhiqiang Zhao,[1] Wan Zhou,[1] and Yuan Mei[2]

[1]*School of Information Science and Technology, Jiujiang University, Jiujiang 332005, China*
[2]*School of Computer, Wuhan University, Wuhan 430072, China*

Correspondence should be addressed to Caixue Zhou; charlesjjjx@126.com

Received 23 March 2017; Revised 15 May 2017; Accepted 22 June 2017; Published 2 August 2017

Academic Editor: Jiankun Hu

Generalized signcryption (GSC) can be applied as an encryption scheme, a signature scheme, or a signcryption scheme with only one algorithm and one key pair. A key-insulated mechanism can resolve the private key exposure problem. To ensure the security of cloud storage, we introduce the key-insulated mechanism into GSC and propose a concrete scheme without bilinear pairings in the certificateless cryptosystem setting. We provide a formal definition and a security model of certificateless key-insulated GSC. Then, we prove that our scheme is confidential under the computational Diffie-Hellman (CDH) assumption and unforgeable under the elliptic curve discrete logarithm (EC-DL) assumption. Our scheme also supports both random-access key update and secure key update. Finally, we evaluate the efficiency of our scheme and demonstrate that it is highly efficient. Thus, our scheme is more suitable for users who communicate with the cloud using mobile devices.

1. Introduction

With the rapid development of cloud storage technology, its security has become increasingly important. For cloud storage, confidentiality and authentication are the two main aspects of security that must be addressed. In general, confidentiality can be realized by encryption and authentication via signature. When both methods are needed simultaneously, the sign-then-encrypt approach is traditionally utilized. However, in this traditional method, the computational costs and communication overhead are the sum of those of signature and encryption, being very high. Signcryption [1] can realize both confidentiality and authentication simultaneously in a single logic step, and the cost is much lower than that of the traditional approach.

Zheng's original signcryption scheme [1] is based on the public key infrastructure (PKI), whose drawback is the high management cost of the public key certificate. An identity-based cryptosystem [2] can greatly reduce the cost of public key management, but it suffers from the private key escrow problem (i.e., the trusted third-party private key generator [PKG] knows all users' private keys).

In 2003, Al-Riyami and Paterson [3] proposed the certificateless cryptosystem, for which a user's private key is composed of two parts: the partial-private key produced by the trusted third-party key generation center (KGC) and a secret value chosen by the user. Accordingly, a user's public key is also composed of two parts: the user's identity information and the public key corresponding to the secret value. Because the public key does not require a certificate, the cost of public key management is greatly reduced. Meanwhile, the KGC does not know the user's secret value, and thus, there is no private key escrow problem. Certificateless cryptosystems have attracted widespread attention since their introduction.

Returning to cloud storage, we sometimes need confidentiality and authentication separately, whereas, at other times, both are needed simultaneously. For example, an announcement needs only authentication, private information requires only confidentiality, and information sent to others needs both. To meet this requirement, we can use three algorithms (i.e., encryption, signature, and signcryption). However, three algorithms require three pairs of keys, and hence, the key management cost is high. To reduce the complexity of key management and increase

the flexibility of implementation, Han et al. [4] proposed the concept of generalized signcryption (GSC) in 2006, which is the natural extension of signcryption. GSC can realize encryption, signature, and signcryption with only one algorithm and one key pair. Because it can adaptively switch between encryption mode, signature mode, and signcryption mode, GSC can realize confidentiality and authentication separately/simultaneously in an efficient manner. Therefore, we can use GSC to achieve confidentiality and authentication separately/simultaneously in a cloud storage scenario.

In Han et al.'s original GSC scheme [4], however, the private key exposure problem was not considered. With the widespread use of mobile devices, key exposure has become a serious and realistic threat. If an attacker can obtain a victim's private key, he/she does not need to solve the hard problems on which the cryptosystem is based. To minimize the damage caused by private key exposure, forward-secure [5], key-insulated [6], and intrusion-resilient [7] technologies have been introduced. In forward-secure technology, the lifetime of a system is divided into separate time periods. At the beginning of each time period, a user's private key must be updated, whileas the public key remains unchanged throughout the lifetime. Thus, the private key exposure of a given time period affects only the security of the current and later time periods, while the previous time periods are protected. Thus, the attacker cannot decrypt the encryption and signcryption ciphertexts in the previous time periods and also cannot forge the signature and signcryption ciphertexts in the previous time periods. As forward-secure technology does not provide backward security, key-insulated technology has been introduced. In this technology, a physically secure but computationally limited device (the helper) is introduced and stores a helper private key. The lifetime of a system is divided into separate time periods, as in forward-security technology. When a user's private key is updated at the beginning of a time period, the helper is needed to produce an update key. The user updates his/her private key by combining the update key and his/her old time period private key. The public key remains unchanged throughout the lifetime, as in the forward-secure technology. The signature, signcryption, decryption, and un-signcryption computations need only the user's current time period private key, and the helper is not involved. Thus, a user's private key being compromised in some given time periods does not affect the security of other time periods. Therefore, both forward and backward security are achieved. In addition, if the helper's private key is exposed, as long as the user's period private key in each time period is not exposed, security will be guaranteed. However, if the helper's private key is exposed at the same time that the user's private key in any time period is exposed, both forward and backward security will be breached. In intrusion-resilient technology, at the beginning of each time period, both the user's private key and the helper's private key must be updated. Thus, it retains the advantage of a key-insulated system while gaining the following advantages. If the helper's private key is exposed in a given time period, both forward and backward security will still hold as long as the user's private key is not exposed within the same time period. If the helper's private key and the user's private key are exposed in the same time period, forward security will be maintained. Obviously, key-insulated technology is more secure than forward-security technology, and intrusion-resilient technology is more secure than key-insulated technology. Due to the complexity and low efficiency of the current intrusion-resilient technology, here, we consider only key-insulated technology.

In this paper, we introduce key-insulated technology into GSC for the first time and propose a certificateless key-insulated GSC scheme to meet the security requirements of cloud storage. Our scheme has the following advantages. First, we use only one algorithm and one key pair to realize the key-insulated encryption, key-insulated signature, and key-insulated signcryption functions. The algorithm can switch between the encryption, signature, and signcryption modes adaptively. Therefore, it can realize confidentiality and authentication separately or simultaneously, and the total number of keys in the system is greatly reduced. Second, the user's private keys may be exposed during some time periods, whereas they are not affected in other time periods. Third, our scheme possesses the advantages of a certificateless cryptosystem: low public key management costs and no private key escrow problem. Fourth, our scheme does not rely on costly bilinear pairings. Bilinear pairing is a useful tool in the design of cryptography schemes, but the computational cost of a pairing can be almost 20 times that of elliptic curve point multiplication [8]. Therefore, the computational efficiency of our scheme is high. Fifth, our scheme supports unbounded time periods. In comparison, in the first key-insulated scheme [6], the total number of time periods must be given in advance. Sixth, our scheme supports random-access key update; that is, for any current time period i and any desired time period j, the private key can be updated from sk_i to sk_j in one step. Seventh, our scheme supports secure key update. We considered the possibility that an adversary may break into the user's storage while a key update is occurring. In this scenario, a key update exposure from time period i to j is equivalent to key exposures in time periods i and j. Other time periods remain secure.

We give a formal definition and the security concept of certificateless key-insulated GSC. Based on the CDH hard problem, we prove that our scheme is confidential in both encryption and signcryption modes. Based on the EC-DL hard problem, we prove that our scheme is unforgeable in both signature and signcryption modes. Finally, we evaluate the efficiency of our scheme and demonstrate that it is highly efficient.

The remainder of this paper is organized as follows. In Section 2, various related works are described. Section 3 addresses various hard problems. In Section 4, the formal definition and security model are introduced. Section 5 presents the concrete certificateless key-insulated GSC scheme without bilinear pairings. In Section 6, we analyze the security of our scheme. In Section 7, we evaluate the efficiency of our scheme. We conclude the paper in Section 8.

2. Related Work

Han et al. [4] first introduced the notion of GSC in 2006. Subsequently, Han and Gui [9] described a multireceiver GSC scheme and applied it to wireless multicast communication in 2009. Wang et al. [10] gave a formal definition and security model of GSC in the PKI setting for the first time and improved the scheme [4] in 2010. Later, Yu et al. [11] proposed an identity-based GSC scheme and a security model in the same year. Kushwah and Lal [12] simplified the security model of scheme [11] and proposed a more efficient identity-based GSC scheme in 2011. Zhou et al. [13] proposed a certificateless GSC scheme that can resist a malicious-but-passive KGC attack [14] in 2014. Wei et al. [15] proposed an identity-based GSC scheme in the standard model and applied it to big data security in 2015. Zhou [16] described an attack on scheme [9] and improved it in the same year. Subsequently, Han and Lu [17] proposed an attribute-based GSC scheme in the standard model and applied it to online social networks. Zhou et al. [18, 19] extended GSC, introduced two new concepts (generalized proxy signcryption and generalized ring signcryption), and proposed a concrete scheme in 2016. Zhang et al. [20] proposed a lightweight certificateless GSC scheme and applied it to a mobile health system in 2017.

Key-insulated encryption was first introduced by Dodis et al. [6] in 2002. Dodis et al. [21] extended the key-insulated encryption to a key-insulated signature and proposed a concrete scheme in 2003. However, in the two schemes, the total number of time periods must be given in advance. Hanaoka et al. [22] introduced key-insulated encryption into the identity-based setting and proposed an identity-based hierarchical key-insulated encryption scheme in 2005. Zhou et al. [23] proposed the first identity-based key-insulated signature in 2006. Weng et al. [24] developed an identity-based key-insulated signature in the standard model in the same year. Subsequently, Hanaoka et al. [25] introduced parallel key-insulated encryption and proposed some concrete schemes. Later, Bellare and Palacio [26] proposed a key-insulated encryption scheme in which the total number of time periods does not need to be determined in advance in the PKI setting. Li et al. [27] introduced the key-insulated mechanism into the group signature and proposed a concrete scheme in 2007. Liu and Wong [28] introduced the key-insulated mechanism into the ring signature and proposed a concrete scheme in 2008. Wan et al. [29] proposed the first certificateless key-insulated signature in 2009; their scheme was designed in the standard model. In the same year, Liu and Cao [30] noted that scheme [24] is insecure. Later, Wan et al. [31] introduced a key-insulated mechanism into the proxy signature and proposed a concrete scheme. Du et al. [32] proposed the first certificate-based key-insulated signature in 2012. Chen et al. [33] proposed the first key-insulated signcryption in the same year and proved their scheme in the standard model. Fan et al. [34] proposed a PKI-based key-insulated signcryption scheme, and Wang et al. [35] suggested an identity-based key-insulated signcryption scheme in 2013. Zhao et al. [36] introduced the key-insulated mechanism into the aggregate signature and proposed a concrete scheme in 2014. Chen et al. [37] proposed the first attribute-based key-insulated signature and applied it to anonymous authentication for a bidirectional broadcasting service in the same year. Subsequently, Zhu et al. [38] determined that scheme [33] is insecure and reported an improvement. Li et al. [39] proposed a certificate-based key-insulated signature scheme in the same year. Xiong et al. [40] proposed a pairing-free certificate-based key-insulated signature scheme for low-power devices, and Lu et al. [41] proposed a certificateless strong key-insulated signature in the standard model in 2015. Li et al. [42] proposed a certificate-based key-insulated signature in the standard model in 2016. Hong and Sun [43] proposed an attribute-based key-insulated signcryption without bilinear pairings and applied it to mobile networks during the same year.

3. Preliminaries

(1) Elliptic curve discrete logarithm (EC-DL) problem: let E be an elliptic curve over the finite field F_p, where p is a prime number, and let G_1 be an additive group of prime order q on $E(F_p)$. Given $(P, aP) \in G_1^2$ for unknown randomly chosen $a \in Z_q$, one must compute a.

(2) Computational Diffie-Hellman (CDH) problem: given $(P, aP, bP) \in G_1^3$ for unknown randomly chosen $a, b \in Z_q$, one must compute abP.

4. Formal Definition and Security Model of Certificateless Key-Insulated GSC

4.1. Formal Definition. A certificateless key-insulated GSC scheme consists of the following eight algorithms.

(1) Setup. Given a security parameter 1^k, it produces a master private key s and a global public parameter Params. It is usually run by the KGC.

(2) Partial-Private-Key-Gen. Given a user's identity ID, the Params, and the master private key s, it produces a partial private key D_{ID} for the user. It is also usually run by the KGC, and the KGC sends D_{ID} to the user securely.

(3) User-Key-Gen. Given a user's identity ID and the Params, it produces a secret value x_{ID} and the corresponding public key $\mathrm{PK}_{\mathrm{ID}}$ for the user. It is usually run by the user.

(4) Set-Initial-Key. Given a user's identity ID, the Params, his/her partial private key D_{ID}, and his/her secret value x_{ID}, it produces a helper private key $\mathrm{hk}_{\mathrm{ID}}$ for the helper and a period private key $S_{\mathrm{ID},0}$ in time period 0 for the user. It is usually run by the user. Then, the user sends the helper private key $\mathrm{hk}_{\mathrm{ID}}$ to the helper and deletes it from the user.

(5) Key-Update-H. Given a user's identity ID, the Params, the helper private key $\mathrm{hk}_{\mathrm{ID}}$, the old time period t, and the new time period t', it produces an update key $\mathrm{UK}_{\mathrm{ID},t,t'}$. It is usually run in the helper device.

(6) Key-Update-U. Given a user's identity ID, the Params, the update key $\mathrm{UK}_{\mathrm{ID},t,t'}$, and a user's period private key $S_{\mathrm{ID},t}$,

it produces the user's period private key $S_{\mathrm{ID},t'}$. It is usually run by the user.

(7) GSC. Given a sender's identity ID_s, a receiver's identity ID_r, the Params, a message m, a time period t, and the sender's period private key $S_{\mathrm{ID}_s,t}$, it produces a GSC ciphertext σ. It is usually run by the sender ID_s or anyone in encryption mode.

This algorithm can be run in three modes.

(a) *Encryption mode:* if ID_s is null and ID_r is not, then the GSC ciphertext σ is an encryption ciphertext.

(b) *Signature mode:* if ID_r is null and ID_s is not, then the GSC ciphertext σ is a signature.

(c) *Signcryption mode:* if neither ID_s nor ID_r is null, then the GSC ciphertext σ is a signcryption ciphertext.

(8) Un-GSC. Given a sender's identity ID_s, a receiver's identity ID_r, the Params, a GSC ciphertext σ, a time period t, and the receiver's period private key $S_{\mathrm{ID}_r,t}$, it recovers the message m in encryption or signcryption mode or returns true in signature mode; otherwise, it returns $\perp$, indicating decryption failure or an invalid signature. It is usually run by the receiver ID_r or anyone in signature mode.

This algorithm can also be run in three modes.

(a) *Decryption mode:* if ID_s is null and ID_r is not, it runs in this mode.

(b) *Signature verification mode:* if ID_r is null and ID_s is not, it runs in this mode. Any person can verify the signature σ.

(c) *Un-signcryption mode:* if neither ID_s nor ID_r is null, it runs in this mode.

Note. The scheme can switch between different modes automatically. If the input of the sender's identity ID_s is null and the receiver's identity ID_r is not, it automatically runs in encryption mode. If the input of ID_s is not null and ID_r is, it automatically runs in signature mode. If the input of neither ID_s nor ID_r is null, it automatically runs in signcryption mode. Both ID_s and ID_r being null is disallowed.

4.2. Security Model. There are two types of attackers in the certificateless cryptosystem [3]. A type I attacker A_{I} does not know the system master private key, but he/she can replace the public key of any user. This considers an attack by any user other than the KGC. A type II attacker A_{II} knows the system master private key, but he/she cannot replace anyone's public key. This considers the attack launched by the KGC, but the KGC is honest-but-curious. Thus, the KGC generates the system parameters honestly according to the Setup algorithm. Then, he/she attempts to attack the system. In 2007, Au et al. [14] introduced a new type of KGC attack: the malicious-but-passive KGC attack. This type of KGC attack assumes that the KGC may imbed some trapdoors in the system parameters when he/she runs the Setup stage. We consider this type of KGC attack in our security model.

In terms of key-insulated security, there are three types of key exposures [6]: (1) ordinary key exposure, which involves the user-period-private key being compromised; (2) key-update exposure, which involves the user's device being compromised during the key-updating step; and (3) helper-key exposure, which involves the physically secure device being compromised. Security against the first type of exposure is called basic key-insulated security, security against the second type of exposure is called secure key update, and security against the last type of exposure is called strong key-insulated security. Compromising both the user-period-private key and helper key is not allowed because, in this case, the adversary can compute the user-period-private key in any time period.

The security of GSC includes confidentiality and unforgeability. Specifically, the scheme must possess indistinguishability under an adaptively chosen ciphertext attack in encryption and signcryption modes and unforgeability under an adaptively chosen message attack in signature and signcryption modes.

By referring to the security models of schemes [6, 13, 33, 34, 39, 41], we have the following nine definitions. The first four definitions focus on the first type of exposure, Definitions 5–8 focus on the last type of exposure, and Definition 9 focuses on the second type of exposure.

There are nine oracles that can be accessed by adversary A as follows.

(a) User-Creation Query. Adversary A provides an identity ID. If it has been created, B returns the public key $\mathrm{PK}_{\mathrm{ID}}$ to A. Otherwise, B runs the partial-key-gen and user-key-gen algorithms to produce the partial private key D_{ID} and the user's secret-value/public key pair $(x_{\mathrm{ID}}, \mathrm{PK}_{\mathrm{ID}})$. B runs the set-initial-key algorithm to produce the helper private key $\mathrm{hk}_{\mathrm{ID}}$. Then, B returns the public key $\mathrm{PK}_{\mathrm{ID}}$ to A.

(b) Partial-Private-Key Query. A provides a created identity ID, and B returns the partial private key D_{ID} to A.

(c) Secret-Value Query. A provides a created identity ID, and B returns the secret value x_{ID} to A.

(d) Public-Key Query. A provides a created identity ID, and B returns the public key $\mathrm{PK}_{\mathrm{ID}}$ to A.

(e) Public-Key-Replacement Query. A provides a created identity ID and a new public/secret-value pair $(\mathrm{PK}'_{\mathrm{ID}}, x'_{\mathrm{ID}})$, and B replaces the old public/secret-value pair $(\mathrm{PK}_{\mathrm{ID}}, x_{\mathrm{ID}})$ with the new one $(\mathrm{PK}'_{\mathrm{ID}}, x'_{\mathrm{ID}})$.

(f) User-Period-Private-Key Query. A provides a created identity ID and a time period t, and B returns the user's period private key $S_{\mathrm{ID},t}$ to A (first running the key-update-h and key-update-u algorithms, if necessary).

(g) Helper-Key Query. A provides a created identity ID. B returns the user's helper key $\mathrm{hk}_{\mathrm{ID}}$ to A.

(h) GSC Query. A provides two created identities $\{\mathrm{ID}_s, \mathrm{ID}_r\}$ (one of them may be null), a message m, and a time period t. B runs the GSC algorithm and returns its output σ to A.

(i) Un-GSC Query. A provides two created identities $\{\mathrm{ID}_s, \mathrm{ID}_r\}$ (one of them may be null), a ciphertext σ, and a time period t. B runs the Un-GSC algorithm and returns its results to A.

4.2.1. Basic Key-Insulated Security

Definition 1 (type I confidentiality, encryption, and signcryption modes). A certificateless key-insulated GSC scheme is said to be indistinguishability-certificateless-basic key-insulated-GSC-adaptive chosen ciphertext attack-type I (IND-CL-Basic-KI-GSC-CCA2-I) secure if no probabilistic polynomial time (PPT) adversary A_{I} has a nonnegligible advantage in the following game.

(1) Setup. Given a security parameter 1^k, challenger B runs the setup algorithm to produce the system public parameter Params and a master private key s. He/she returns Params to adversary A_{I} and keeps s secret.

(2) Find Stage. A_{I} can adaptively ask all the above oracles, except the helper-key oracle.

(3) Challenge Stage. A_{I} provides two distinct messages $\{m_0, m_1\}$ with equal length, a created sender's identity ID_s^* (ID_s^* may be null), a created receiver's identity ID_r^*, and a time period t^*. B randomly chooses $b \in \{0, 1\}$ and computes $\sigma^* = \mathrm{GSC}(m_b, \mathrm{Params}, S_{\mathrm{ID}_s^*, t^*}, \mathrm{PK}_{\mathrm{ID}_r^*})$. Then, B returns σ^* to A_{I}.

(4) Guess Stage. A_{I} can ask the same queries as in the Find stage adaptively. Finally, A_{I} gives his/her guess b'. If $b' = b$, he/she wins the game. The restrictions on A_{I} are as follows:

(a) The receiver's identity ID_r^* cannot be null.

(b) A_{I} cannot ask for ID_r^*'s period private key $S_{\mathrm{ID}_r^*, t^*}$ in time period t^*.

(c) A_{I} cannot ask for the partial private key $D_{\mathrm{ID}_r^*}$ of ID_r^*.

(d) In the Guess stage, A_{I} cannot make an Un-GSC query on the challenge ciphertext σ^* under ID_s^*, ID_r^*, and t^* unless the public key of ID_s^* or ID_r^* has been replaced after the Challenge stage.

A_{I}'s advantage is defined as $\mathrm{Adv}_{A_{\mathrm{I}}}^{\text{IND-CLKI-GSC-CCA2}} = 2\Pr[b' = b] - 1$.

Note. In the above Challenge stage, the sender's identity ID_s^* may be null. In this case, it runs in encryption mode; otherwise it runs in signcryption mode. Thus, the encryption and signcryption modes share the same game, as described above.

Definition 2 (type II confidentiality, encryption, and signcryption modes). A certificateless key-insulated GSC scheme is said to be IND-CL-Basic-KI-GSC-CCA2-II secure if no PPT adversary A_{II} has a nonnegligible advantage in the following game.

(1) Setup. Given a security parameter 1^k, adversary A_{II} runs the setup algorithm to produce the system public parameter Params and a master private key s. He/she returns Params and s to challenger B.

(2) Find Stage. A_{II} can adaptively ask all the above oracles except the helper-key, partial-private-key, and public-key-replacement oracles. A_{II} holds the master private key and thus he/she can compute the partial private key by himself/herself. A_{II} is not allowed to replace anyone's public key in the certificateless cryptographic system.

(3) Challenge Stage. A_{II} provides two distinct messages $\{m_0, m_1\}$ with equal length, a created sender's identity ID_s^* (ID_s^* may be null), a created receiver's identity ID_r^*, and a time period t^*. B randomly chooses $b \in \{0, 1\}$ and computes $\sigma^* = \mathrm{GSC}(m_b, \mathrm{Params}, S_{\mathrm{ID}_s^*, t^*}, \mathrm{PK}_{\mathrm{ID}_r^*})$. Then, B returns σ^* to A_{II}.

(4) Guess Stage. A_{II} can ask the same queries adaptively as in the Find stage. Finally, A_{II} gives his/her guess b'. If $b' = b$, he/she wins the game. The restrictions on A_{II} are as follows:

(a) The receiver's identity ID_r^* cannot be null.

(b) A_{II} cannot ask for ID_r^*'s period private key $S_{\mathrm{ID}_r^*, t^*}$ in time period t^*.

(c) A_{II} cannot ask for ID_r^*'s secret value $x_{\mathrm{ID}_r^*}$

(d) In the Guess stage, A_{II} cannot make an Un-GSC query on the challenge ciphertext σ^* under ID_s^*, ID_r^*, and t^*.

A_{II}'s advantage is defined as $\mathrm{Adv}_{A_{\mathrm{II}}}^{\text{IND-CLKI-GSC-CCA2}} = 2\Pr[b' = b] - 1$.

Note 1. In the above Challenge stage, the sender's identity ID_s^* may be null. In this case, it runs in encryption mode; otherwise, it runs in signcryption mode. Thus, the encryption and signcryption modes share the same game, as described above.

Note 2. To resist the malicious-but-passive KGC attack, it must let adversary A_{II} produce the system parameters Params and master private key s in the Setup stage.

Definition 3 (type I unforgeability, signature, and signcryption modes). A certificateless key-insulated GSC scheme is said to be existentially unforgeable-certificateless-basic key-insulated-GSC-adaptive chosen message attack-type I (EUF-CL-Basic-KI-GSC-CMA-I) secure if no PPT adversary A_{I} has a nonnegligible advantage in the following game.

(1) Setup. The same as in Definition 1.

(2) Queries. The same as in Definition 1.

(3) Forgery. Finally, A_{I} outputs a forged GSC ciphertext σ^* in time period t^* with ID_s^* as the sender and ID_r^* as the receiver.

A_{I} wins the game if the output of the Un-GSC algorithm is not the symbol $\perp$ and if the following conditions hold:

(a) The sender's identity ID_s^* cannot be null.

(b) A_{I} cannot ask for ID_s^*'s period private key $S_{\mathrm{ID}_s^*,t^*}$ in time period t^*.

(c) A_{I} cannot ask for ID_s^*'s partial private key $D_{\mathrm{ID}_r^*}$.

(d) $(\sigma^*, t^*, \mathrm{ID}_s^*, \mathrm{ID}_r^*)$ is not the output of the GSC query.

A_{I}'s advantage is its probability of victory.

Note. In the above Forgery stage, the receiver's identity ID_r^* may be null. In this case, it runs in signature mode; otherwise, it runs in signcryption mode. Thus, the signature and signcryption modes share the same game, as described above.

Definition 4 (type II unforgeability, signature, and signcryption modes). A certificateless key-insulated GSC scheme is said to be EUF-CL-Basic-KI-GSC-CMA-II secure if no PPT adversary A_{II} has a nonnegligible advantage in the following game.

(1) Setup. The same as in Definition 2.

(2) Queries. The same as in Definition 2.

(3) Forgery. Finally, A_{II} outputs a forged GSC ciphertext σ^* in time period t^* with ID_s^* as the sender and ID_r^* as the receiver. A_{II} wins the game if the output of the Un-GSC algorithm is not the symbol $\perp$ and if the following conditions hold:

(a) The sender's identity ID_s^* cannot be null.

(b) A_{II} cannot ask for ID_s^*'s period private key $S_{\mathrm{ID}_s^*,t^*}$ in time period t^*.

(c) A_{II} cannot ask for ID_s^*'s secret value $x_{\mathrm{ID}_s^*}$.

(d) $(\sigma^*, t^*, \mathrm{ID}_s^*, \mathrm{ID}_r^*)$ is not the output of the GSC query.

A_{II}'s advantage is its probability of victory.

Note. In the above Forgery stage, the receiver's identity ID_r^* may be null. In this case, it runs in signature mode; otherwise, it runs in signcryption mode. Thus, the signature and signcryption modes share the same game, as described above.

4.2.2. Strong Key-Insulated Security

Definition 5 (type I confidentiality, encryption, and signcryption modes). A certificateless key-insulated GSC scheme is said to be IND-CL-Strong-KI-GSC-CCA2-I secure if no PPT adversary A_{I} has a nonnegligible advantage in the following game.

(1) Setup. The same as in Definition 1.

(2) Find Stage. A_{I} can adaptively ask all the above oracles except the user-period-private-key oracle.

(3) Challenge Stage. The same as in Definition 1.

(4) Guess Stage. A_{I} can adaptively make the same queries as in the Find stage. Finally, A_{I} gives his/her guess b'. If $b' = b$, he/she wins the game. The restrictions on A_{I} are as follows:

(a) The receiver's identity ID_r^* cannot be null.

(b) A_{I} cannot ask for the partial private key $D_{\mathrm{ID}_r^*}$ of ID_r^*.

(c) In the Guess stage, A_{I} cannot make an Un-GSC query on the challenge ciphertext σ^* under ID_s^*, ID_r^*, and t^* unless the public key of ID_s^* or ID_r^* has been replaced after the challenge stage.

A_{I}'s advantage is defined as $\mathrm{Adv}_{A_{\mathrm{I}}}^{\text{IND-CLKI-GSC-CCA2}} = 2\Pr[b' = b] - 1$.

Note. In the above Challenge stage, the sender's identity ID_s^* may be null. In this case, it runs in encryption mode; otherwise, it runs in signcryption mode. Thus, the encryption and signcryption modes share the same game, as described above.

Definition 6 (type II confidentiality, encryption, and signcryption modes). A certificateless key-insulated GSC scheme is said to be IND-CL-Strong-KI-GSC-CCA2-II secure if no PPT adversary A_{II} has a nonnegligible advantage in the following game.

(1) Setup. The same as in Definition 2.

(2) Find Stage. A_{II} can adaptively ask all the above oracles except the user-period-private-key, partial-private-key, and public-key-replacement oracles.

(3) Challenge Stage. The same as in Definition 2.

(4) Guess Stage. A_{II} can make the same queries adaptively as in the Find stage. Finally, A_{II} gives his/her guess b'. If $b' = b$, he/she wins the game. The restrictions on A_{II} are as follows:

(a) The receiver's identity ID_r^* cannot be null.

(b) A_{II} cannot ask for ID_r^*'s secret value $x_{\mathrm{ID}_r^*}$.

(c) In the Guess stage, A_{II} cannot make an Un-GSC query on the challenge ciphertext σ^* under ID_s^*, ID_r^*, and t^*.

A_{II}'s advantage is defined as $\mathrm{Adv}_{A_{\mathrm{II}}}^{\text{IND-CLKI-GSC-CCA2}} = 2\Pr[b' = b] - 1$.

Note 1. In the above Challenge stage, the sender's identity ID_s^* may be null. In this case, it runs in encryption mode; otherwise, it runs in signcryption mode. Thus, the encryption and signcryption modes share the same game, as described above.

Note 2. To resist the malicious-but-passive KGC attack, it must let adversary A_{II} produce the system parameters Params and master private key s in the Setup stage.

Definition 7 (type I unforgeability, signature, and signcryption modes). A certificateless key-insulated GSC scheme is said to be EUF-CL-Strong-KI-GSC-CMA-I secure if no PPT adversary A_I has a nonnegligible advantage in the following game.

(1) Setup. The same as in Definition 5.

(2) Queries. The same as in Definition 5.

(3) Forgery. Finally, A_I outputs a forged GSC ciphertext σ^* in time period t^* with ID_s^* as the sender and ID_r^* as the receiver. A_I wins the game if the output of the Un-GSC algorithm is not the symbol $\perp$ and if the following conditions hold:

(a) The sender's identity ID_s^* cannot be null.

(b) A_I cannot ask for ID_s^*'s partial private key $D_{ID_r^*}$.

(c) $(\sigma^*, t^*, ID_s^*, ID_r^*)$ is not the output of the GSC query.

A_I's advantage is its probability of victory.

Note. In the above Forgery stage, the receiver's identity ID_r^* may be null. In this case, it runs in signature mode; otherwise, it runs in signcryption mode. Thus, the signature and signcryption modes share the same game, as described above.

Definition 8 (type II unforgeability, signature, and signcryption modes). A certificateless key-insulated GSC scheme is said to be EUF-CL-Strong-KI-GSC-CMA-II secure if no PPT adversary A_{II} has a nonnegligible advantage in the following game.

(1) Setup. The same as in Definition 6.

(2) Queries. The same as in Definition 6.

(3) Forgery. Finally, A_{II} outputs a forged GSC ciphertext σ^* in time period t^* with ID_s^* as the sender and ID_r^* as the receiver. A_{II} wins the game if the output of the Un-GSC algorithm is not the symbol $\perp$ and if the following conditions hold:

(a) The sender's identity ID_s^* cannot be null.

(b) A_{II} cannot ask for ID_s^*'s secret value $x_{ID_s^*}$.

(c) $(\sigma^*, t^*, ID_s^*, ID_r^*)$ is not the output of the GSC query.

A_{II}'s advantage is its probability of victory.

Note. In the above Forgery stage, the receiver's identity ID_r^* may be null. In this case, it runs in signature mode; otherwise, it runs in signcryption mode. Thus, the signature and signcryption modes share the same game, as described above.

4.2.3. Secure Key Update

Definition 9. A certificateless key-insulated GSC scheme is said to support secure key update if a key update exposure from time period t to t' is equivalent to user-period-private-key exposures in both time periods t and t'.

5. A Concrete Certificateless Key-Insulated GSC Scheme

5.1. Concrete Scheme. By referring to Seo et al.'s [44] pairing-free certificateless signcryption tag key encapsulation mechanism, we propose an efficient pairing-free certificateless key-insulated GSC scheme.

Setup. Given a security parameter 1^k, the KGC produces two large prime numbers p and q. Then, he/she defines a secure elliptic curve $E(F_p)$ on the finite field F_p. Let G_1 be a cyclic group of order q on $E(F_p)$, and let P be a generator of G_1. He/she randomly chooses $s \in Z_q^*$ as the master private key and computes $P_{pub} = s \cdot P$ as the master public key. He/she chooses seven hash functions: $H_0, H_1, H_2, H_3, H_4, H_5 : \{0,1\}^* \to Z_q^*$, and $H_6 : \{0,1\}^* \to \{0,1\}^L \times z_q^*$, where L is the bit length of message m. He/she defines a special function $f(ID)$: if the identity ID is null, then $f(ID) = 0$; otherwise, $f(ID) = 1$. The system public parameters are $Params = \{p, q, E(F_p), G_1, P, P_{pub}, f, H_0, H_1, H_2, H_3, H_4, H_5, H_6\}$. He/she keeps the master private key s secret.

Partial-Private-Key-Gen. Given a user's identity ID, the KGC randomly chooses $r_{ID} \in Z_q^*$ and computes $Y_{ID} = r_{ID} \cdot P$ and $y_{ID} = r_{ID} + s \cdot h_{0,ID} \bmod q$, where $h_{0,ID} = H_0(ID, Y_{ID})$. KGC sends y_{ID} to ID securely.

User-Key-Gen. The user with identity ID randomly chooses $x_{ID} \in Z_q^*$ as his/her secret value. He/she computes the public key as $X_{ID} = x_{ID} \cdot P$.

Set-Initial-Key. The user with identity ID randomly chooses $u_{ID,0}, hk_{ID} \in Z_q^*$ and computes $U_{ID,0} = u_{ID,0} \cdot P$, $T_{ID} = hk_{ID} \cdot P$, $h_{1,ID,0} = H_1(ID, Y_{ID}, T_{ID}, 0)$, $h_{2,ID} = H_2(ID, Y_{ID}, X_{ID}, T_{ID})$, $h_{3,ID,0} = H_3(ID, Y_{ID}, U_{ID,0}, 0)$, and $s_{ID,0} = y_{ID} + x_{ID} \cdot h_{2,ID} + u_{ID,0} \cdot h_{3,ID,0} + hk_{ID} \cdot h_{1,ID,0} \bmod q$. Finally, the helper private key is hk_{ID}, and the user's period private key in time period 0 is $s_{ID,0}$. The user broadcasts $(U_{ID,0}, Y_{ID}, T_{ID})$. Then, the user sends the helper key hk_{ID} and the ephemeral variable $u_{ID,0}$ in time period 0 to the helper and deletes them from the user.

Key-Update-H. Given a user's identity ID and Y_{ID}, the old time period t, and the new time period t', the helper chooses $u_{ID,t'} \in Z_q^*$ and computes $U_{ID,t'} = u_{ID,t'} \cdot P$, $U_{ID,t} = u_{ID,t} \cdot P$,

$T_{\mathrm{ID}} = \mathrm{hk}_{\mathrm{ID}} \cdot P$, $h_{1,\mathrm{ID},t'} = H_1(\mathrm{ID}, Y_{\mathrm{ID}}, T_{\mathrm{ID}}, t')$, $h_{1,\mathrm{ID},t} = H_1(\mathrm{ID}, Y_{\mathrm{ID}}, T_{\mathrm{ID}}, t)$, $h_{3,\mathrm{ID},t'} = H_3(\mathrm{ID}, Y_{\mathrm{ID}}, U_{\mathrm{ID},t'}, t')$, $h_{3,\mathrm{ID},t} = H_3(\mathrm{ID}, Y_{\mathrm{ID}}, U_{\mathrm{ID},t}, t)$, and $\mathrm{uk}_{\mathrm{ID},t,t'} = u_{\mathrm{ID},t'} \cdot h_{3,\mathrm{ID},t'} - u_{\mathrm{ID},t} \cdot h_{3,\mathrm{ID},t} + \mathrm{hk}_{\mathrm{ID}} \cdot (h_{1,\mathrm{ID},t'} - h_{1,\mathrm{ID},t}) \bmod q$. Then, the update key is $(\mathrm{uk}_{\mathrm{ID},t,t'}, U_{\mathrm{ID},t'})$. The helper saves $u_{\mathrm{ID},t'}$ and deletes $u_{\mathrm{ID},t}$.

Key-Update-U. Given the update key $(\mathrm{uk}_{\mathrm{ID},t,t'}, U_{\mathrm{ID},t'})$, the user ID updates his/her period private key from time period t to t' as $s_{\mathrm{ID},t'} = s_{\mathrm{ID},t} + \mathrm{uk}_{\mathrm{ID},t,t'}$. Then, he/she broadcasts $U_{\mathrm{ID},t'}$.

GSC. Let $m \in \{0,1\}^L$, let the sender's identity be ID_s, let the receiver's identity be ID_r, and let the time period be t. The sender randomly chooses $a_1, a_2 \in Z_q^*$ and computes $R_1 = a_1 \cdot P$, $R_2 = a_2 \cdot P$, $h_4 = H_4(m, R_1, R_2, \mathrm{ID}_s, Y_{\mathrm{ID}_s}, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$, $h_5 = H_5(m, R_1, R_2, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$, and $u = f(\mathrm{ID}_s) \cdot s_{\mathrm{ID}_s,t} \cdot h_4 + a_1 \cdot h_5 + a_2 \bmod q$. Then, he/she computes $h_{0,\mathrm{ID}_r} = H_0(\mathrm{ID}_r, Y_{\mathrm{ID}_r})$, $h_{1,\mathrm{ID}_r,t} = H_1(\mathrm{ID}_r, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r}, t)$, $h_{2,\mathrm{ID}_r} = H_2(\mathrm{ID}_r, Y_{\mathrm{ID}_r}, X_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$, $h_{3,\mathrm{ID}_r,t} = H_3(\mathrm{ID}_r, Y_{\mathrm{ID}_r}, U_{\mathrm{ID}_r,t}, t)$, $V = a_1 \cdot (Y_{\mathrm{ID}_r} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_r} + X_{\mathrm{ID}_r} \cdot h_{2,\mathrm{ID}_r} + U_{\mathrm{ID}_r,t} \cdot h_{3,\mathrm{ID}_r,t} + T_{\mathrm{ID}_r} \cdot h_{1,\mathrm{ID}_r,t})$, $h_6 = f(\mathrm{ID}_r) \cdot H_6(\mathrm{ID}_s, \mathrm{ID}_r, U_{\mathrm{ID}_s,t}, X_{\mathrm{ID}_s}, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, R_1, V, t)$, and $c = (m \parallel u) \oplus h_6$. Finally, the output is $(t, \sigma) = (t, (R_1, R_2, c))$.

This algorithm can be run in three modes. We add a tag in the ciphertext.

(1) Encryption Mode. If ID_s is null and ID_r is not, then $f(\mathrm{ID}_s) = 0$ and $f(\mathrm{ID}_r) = 1$. The ciphertext $(t, \sigma) = (t, (R_1, R_2, c, \mathrm{tag} = 1))$ is an encryption ciphertext. In this case, $U_{\mathrm{ID}_s,t}$, X_{ID_s}, Y_{ID_s}, and T_{ID_s} are all set to the infinite point ∞ on $E(F_p)$, and $s_{\mathrm{ID}_s,t}$ is set to zero. Additionally, in this case, $u = a_1 \cdot h_5 + a_2 \bmod q$.

(2) Signature Mode. If ID_r is null and ID_s is not, then $f(\mathrm{ID}_r) = 0$ and $f(\mathrm{ID}_s) = 1$. The ciphertext $(t, \sigma) = (t, (R_1, R_2, c, \mathrm{tag} = 2))$ is a signature. In this case, $U_{\mathrm{ID}_r,t}$, X_{ID_r}, Y_{ID_r}, and T_{ID_r} are all set to the infinite point ∞ on $E(F_p)$. Additionally, in this case, $h_6 = 0$ and $c = m \parallel u$.

(3) Signcryption Mode. If neither ID_s nor ID_r is null, then $f(\mathrm{ID}_s) = 1$ and $f(\mathrm{ID}_r) = 1$. The ciphertext $(t, \sigma) = (t, (R_1, R_2, c, \mathrm{tag} = 3))$ is a signcryption ciphertext.

Un-GSC. The ciphertext is $(t, \sigma) = (t, (R_1, R_2, c, \mathrm{tag}))$.

(a) tag = 1. $\sigma = (R_1, R_2, c)$ is an encryption ciphertext. The receiver ID_r computes $V = s_{\mathrm{ID}_r,t} \cdot R_1$ and $h_6 = H_6(\mathrm{null}, \mathrm{ID}_r, \infty, \infty, \infty, \infty, R_1, V, t)$ and recovers the message $m \parallel u = c \oplus h_6$. Then, he/she computes $h_5 = H_5(m, R_1, R_2, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$ and verifies whether $u \cdot P = R_1 \cdot h_5 + R_2$ holds true or not. If it does, he/she accepts it.

(b) tag = 2. $\sigma = (R_1, R_2, c)$ is a signature. In this case, $c = m \parallel u$. The verifier computes $h_{0,\mathrm{ID}_s} = H_0(\mathrm{ID}_s, Y_{\mathrm{ID}_s})$, $h_{1,\mathrm{ID}_s,t} = H_1(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, t)$, $h_{2,\mathrm{ID}_s} = H_2(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, X_{\mathrm{ID}_s}, T_{\mathrm{ID}_s})$, $h_{3,\mathrm{ID}_s,t} = H_3(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, U_{\mathrm{ID}_s,t}, t)$, $h_4 = H_4(m, R_1, R_2, \mathrm{ID}_s, Y_{\mathrm{ID}_s}, \mathrm{null}, \infty, \infty, \infty, \infty)$, and $h_5 = H_5(m, R_1, R_2, \mathrm{null}, \infty, \infty, \infty, \infty)$. Then, he/she verifies whether $u \cdot P = (Y_{\mathrm{ID}_s} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_s} + X_{\mathrm{ID}_s} \cdot h_{2,\mathrm{ID}_s} + U_{\mathrm{ID}_s,t} \cdot h_{3,\mathrm{ID}_s,t} + T_{\mathrm{ID}_s} \cdot h_{1,\mathrm{ID}_s,t}) \cdot h_4 + R_1 \cdot h_5 + R_2$ holds true or not. If it does, he/she accepts the signature.

(c) tag = 3. $\sigma = (R_1, R_2, c)$ is a signcryption ciphertext. The receiver ID_r computes $V = s_{\mathrm{ID}_r,t} \cdot R_1$ and $h_6 = H_6(\mathrm{ID}_s, \mathrm{ID}_r, U_{\mathrm{ID}_s,t}, X_{\mathrm{ID}_s}, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, R_1, V, t)$ and recovers the message $m \parallel u = c \oplus h_6$. Then, he/she computes $h_{0,\mathrm{ID}_s} = H_0(\mathrm{ID}_s, Y_{\mathrm{ID}_s})$, $h_{1,\mathrm{ID}_s,t} = H_1(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, t)$, $h_{2,\mathrm{ID}_s} = H_2(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, X_{\mathrm{ID}_s}, T_{\mathrm{ID}_s})$, $h_{3,\mathrm{ID}_s,t} = H_3(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, U_{\mathrm{ID}_s,t}, t)$, $h_4 = H_4(m, R_1, R_2, \mathrm{ID}_s, Y_{\mathrm{ID}_s}, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$, and $h_5 = H_5(m, R_1, R_2, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$ and verifies whether $u \cdot P = (Y_{\mathrm{ID}_s} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_s} + X_{\mathrm{ID}_s} \cdot h_{2,\mathrm{ID}_s} + U_{\mathrm{ID}_s,t} \cdot h_{3,\mathrm{ID}_s,t} + T_{\mathrm{ID}_s} \cdot h_{1,\mathrm{ID}_s,t}) \cdot h_4 + R_1 \cdot h_5 + R_2$ holds true or not. If it does, he/she accepts it.

5.2. Correctness

①

$$\begin{aligned} u \cdot P &= \left(s_{\mathrm{ID}_s,t} \cdot h_4 + a_1 \cdot h_5 + a_2\right) \cdot P = \Big(\Big(y_{\mathrm{ID}_s} + x_{\mathrm{ID}_s} \\ &\cdot h_{2,\mathrm{ID}_s} + u_{\mathrm{ID}_s,t} \cdot h_{3,\mathrm{ID}_s,t} + \mathrm{hk}_{\mathrm{ID}_s} \cdot h_{1,\mathrm{ID}_s,t}\Big) \cdot h_4 + a_1 \\ &\cdot h_5 + a_2\Big) \cdot P = \Big(\Big(r_{\mathrm{ID}_s} + s \cdot h_{0,\mathrm{ID}_s} + x_{\mathrm{ID}_s} \cdot h_{2,\mathrm{ID}_s} \\ &+ u_{\mathrm{ID}_s,t} \cdot h_{3,\mathrm{ID}_s,t} + \mathrm{hk}_{\mathrm{ID}_s} \cdot h_{1,\mathrm{ID}_s,t}\Big) \cdot h_4 + a_1 \cdot h_5 + a_2\Big) \\ &\cdot P = \left(Y_{\mathrm{ID}_s} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_s} + X_{\mathrm{ID}_s} \cdot h_{2,\mathrm{ID}_s} + U_{\mathrm{ID}_s,t}\right. \\ &\left. \cdot h_{3,\mathrm{ID}_s,t} + T_{\mathrm{ID}_s} \cdot h_{1,\mathrm{ID}_s,t}\right) \cdot h_4 + R_1 \cdot h_5 + R_2. \end{aligned} \tag{1}$$

②

$$\begin{aligned} V &= a_1 \cdot \left(Y_{\mathrm{ID}_r} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_r} + X_{\mathrm{ID}_r} \cdot h_{2,\mathrm{ID}_r} + U_{\mathrm{ID}_r,t}\right. \\ &\left. \cdot h_{3,\mathrm{ID}_r,t} + T_{\mathrm{ID}_r} \cdot h_{1,\mathrm{ID}_r,t}\right) = a_1 \cdot P \cdot \left(r_{\mathrm{ID}_r} + s \cdot h_{0,\mathrm{ID}_r}\right. \\ &\left. + x_{\mathrm{ID}_r} \cdot h_{2,\mathrm{ID}_r} + u_{\mathrm{ID}_r,t} \cdot h_{3,\mathrm{ID}_r,t} + \mathrm{hk}_{\mathrm{ID}_r} \cdot h_{1,\mathrm{ID}_r,t}\right) \\ &= \left(y_{\mathrm{ID}_r} + x_{\mathrm{ID}_r} \cdot h_{2,\mathrm{ID}_r} + u_{\mathrm{ID}_r,t} \cdot h_{3,\mathrm{ID}_r,t} + \mathrm{hk}_{\mathrm{ID}_r}\right. \\ &\left. \cdot h_{1,\mathrm{ID}_r,t}\right) \cdot R_1 = s_{\mathrm{ID}_r,t} \cdot R_1. \end{aligned} \tag{2}$$

5.3. Random-Access Key Update. Obviously, our scheme supports random-access key update; thus, for any current time period i and any desired time period j, the private key can be updated from sk_i to sk_j in one step.

6. Security Analysis of the Proposed Scheme

6.1. Confidentiality of Basic Key Insulation

Theorem 10 (type I confidentiality). *In the random oracle model, if there is a PPT adversary A_I with a nonnegligible advantage ε against the IND-CL-Basic-KI-GSC-CCA2-I security of the scheme running in encryption or signcryption mode in time t and performing at most q_{H_i} H_i $(i = 0, 1, 2, \ldots, 6)$ queries, $q_{U\text{-}C}$ user-creation queries, $q_{Pa\text{-}p\text{-}k}$ partial-private-key*

queries, $q_{Pe\text{-}p\text{-}k}$ period-private-key queries, q_{GSC} GSC queries, and $q_{Un\text{-}GSC}$ Un-GSC queries, then the CDH problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{U\text{-}C} \cdot (1 - 1/q_{U\text{-}C})^{q_{Pa\text{-}p\text{-}k}} \cdot (1 - 1/q_{U\text{-}C})^{q_{Pe\text{-}p\text{-}k}} \cdot (q_{Un\text{-}GSC}/2^k)$ in time $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, where t_m denotes the time for a scalar multiplication on G_1.

Proof. Suppose challenger C is given $(P, aP, bP) \in G_1^3$ for random $a, b \in Z_q^*$. C does not know the values of a and b and is asked to compute abP. To utilize adversary A_{I}, challenger C will simulate all the oracles defined in Definition 1.

Setup. C sets $P_{\mathrm{pub}} = aP$. Other public parameters are produced normally. C gives the system public parameters $\mathrm{Params} = \{p, q, E(F_p), G_1, P, P_{\mathrm{pub}}, f, H_0, H_1, H_2, H_3, H_4, H_5, H_6\}$ to A_{I}. C maintains nine lists, $L_0, L_1, L_2, L_3, L_4, L_5, L_6, L_k$, and $L_{\text{Pe-k}}$, which are initially empty. C randomly selects $\lambda \in \{1, 2, \ldots, q_{\text{U-C}}\}$.

Find Stage. A_{I} makes queries to the following oracles adaptively.

User-Creation Query. A_{I} provides an identity ID. C looks up list L_k to determine whether it contains the item. If it does, C returns ID's public key X_{ID} and public parameters $(Y_{\mathrm{ID}}, T_{\mathrm{ID}})$ to A_{I}. Otherwise, C proceeds as follows and returns public key X_{ID} and public parameters $(Y_{\mathrm{ID}}, T_{\mathrm{ID}})$ to A_{I}.

C randomly selects $x_{\mathrm{ID}} \in Z_q^*$ as the secret value and computes the public key as $X_{\mathrm{ID}} = x_{\mathrm{ID}} \cdot P$. C randomly selects $\mathrm{hk}_{\mathrm{ID}} \in Z_q^*$ as the helper private key and computes $T_{\mathrm{ID}} = \mathrm{hk}_{\mathrm{ID}} \cdot P$.

(1) $\mathrm{ID} = \mathrm{ID}_\lambda$. C randomly selects $r_{\mathrm{ID}_\lambda} \in Z_q^*$ and computes $Y_{\mathrm{ID}_\lambda} = r_{\mathrm{ID}_\lambda} \cdot P$. C inserts the tuple $(\mathrm{ID}_\lambda, X_{\mathrm{ID}_\lambda}, Y_{\mathrm{ID}_\lambda}, x_{\mathrm{ID}_\lambda}, r_{\mathrm{ID}_\lambda}, -, \mathrm{hk}_{\mathrm{ID}_\lambda}, T_{\mathrm{ID}_\lambda})$ into list L_k.

(2) $\mathrm{ID} \neq \mathrm{ID}_\lambda$. C randomly selects $y_{\mathrm{ID}}, h_{0,\mathrm{ID}} \in Z_q^*$ and computes $Y_{\mathrm{ID}} = y_{\mathrm{ID}} \cdot P - h_{0,\mathrm{ID}} \cdot P_{\mathrm{pub}}$. C inserts the tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, h_{0,\mathrm{ID}})$ into list L_0. If there is a collision in list L_0, C rechooses $y_{\mathrm{ID}}, h_{0,\mathrm{ID}} \in Z_q^*$ and repeats the process. C inserts the tuple $(\mathrm{ID}, X_{\mathrm{ID}}, Y_{\mathrm{ID}}, x_{\mathrm{ID}}, -, y_{\mathrm{ID}}, \mathrm{hk}_{\mathrm{ID}}, T_{\mathrm{ID}})$ into list L_k.

H_0 Query. A_{I} supplies a tuple $(\mathrm{ID}, Y_{\mathrm{ID}})$. C first checks list L_0 to determine whether it contains the item $(\mathrm{ID}, Y_{\mathrm{ID}}, h_{0,\mathrm{ID}})$. If it does, C returns $h_{0,\mathrm{ID}}$. Otherwise, C randomly selects $h_{0,\mathrm{ID}} \in Z_q^*$ and repeats the process until $h_{0,\mathrm{ID}}$ is not in list L_0. C stores the tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, h_{0,\mathrm{ID}})$ in list L_0 and returns $h_{0,\mathrm{ID}}$ to A_{I}.

H_1 Query. A_{I} supplies a tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, T_{\mathrm{ID}}, t)$. C first checks list L_1 to determine whether it contains the item $(\mathrm{ID}, Y_{\mathrm{ID}}, T_{\mathrm{ID}}, t, h_{1,\mathrm{ID},t})$. If it does, C returns $h_{1,\mathrm{ID},t}$. Otherwise, C randomly selects $h_{1,\mathrm{ID},t} \in Z_q^*$ and repeats the process until $h_{1,\mathrm{ID},t}$ is not in list L_1. C stores the tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, T_{\mathrm{ID}}, t, h_{1,\mathrm{ID},t})$ in list L_1 and returns $h_{1,\mathrm{ID},t}$ to A_{I}.

H_2 Query. A_{I} supplies a tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, X_{\mathrm{ID}}, T_{\mathrm{ID}})$. C first checks list L_2 to determine whether it contains the item $(\mathrm{ID}, Y_{\mathrm{ID}}, X_{\mathrm{ID}}, T_{\mathrm{ID}}, h_{2,\mathrm{ID}})$. If it does, C returns $h_{2,\mathrm{ID}}$. Otherwise, C randomly selects $h_{2,\mathrm{ID}} \in Z_q^*$ and repeats the process until $h_{2,\mathrm{ID}}$ is not in list L_2. C stores the tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, X_{\mathrm{ID}}, T_{\mathrm{ID}}, h_{2,\mathrm{ID}})$ in list L_2 and returns $h_{2,\mathrm{ID}}$ to A_{I}.

H_3 Query. A_{I} supplies a tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, U_{\mathrm{ID},t}, t)$. C first checks list L_3 to determine whether it contains the item $(\mathrm{ID}, Y_{\mathrm{ID}}, U_{\mathrm{ID},t}, t, h_{3,\mathrm{ID},t})$. If it does, C returns $h_{3,\mathrm{ID},t}$. Otherwise, C randomly selects $h_{3,\mathrm{ID},t} \in Z_q^*$ and repeats the process until $h_{3,\mathrm{ID},t}$ is not in list L_3. C stores the tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, U_{\mathrm{ID},t}, t, h_{3,\mathrm{ID},t})$ in list L_3 and returns $h_{3,\mathrm{ID},t}$ to A_{I}.

H_4 Query. A_{I} supplies a tuple $(m, R_1, R_2, \mathrm{ID}_s, Y_{\mathrm{ID}_s}, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$. C first checks list L_4 to determine whether it contains the item $(m, R_1, R_2, \mathrm{ID}_s, Y_{\mathrm{ID}_s}, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r}, h_4)$. If it does, C returns h_4. Otherwise, C randomly selects $h_4 \in Z_q^*$ and repeats the process until h_4 is not in list L_4. C stores the tuple $(m, R_1, R_2, \mathrm{ID}_s, Y_{\mathrm{ID}_s}, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r}, h_4)$ in list L_4 and returns h_4 to A_{I}.

H_5 Query. A_{I} supplies a tuple $(m, R_1, R_2, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$. C first checks list L_5 to determine whether it contains the item $(m, R_1, R_2, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r}, h_5)$. If it does, C returns h_5. Otherwise, C randomly selects $h_5 \in Z_q^*$ and repeats the process until h_5 is not in list L_5. C stores the tuple $(m, R_1, R_2, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r}, h_5)$ in list L_5 and returns h_5 to A_{I}.

H_6 Query. A_{I} supplies a tuple $(\mathrm{ID}_s, \mathrm{ID}_r, U_{\mathrm{ID}_s,t}, X_{\mathrm{ID}_s}, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, R_1, V, t)$. C first checks list L_6 to determine whether it contains the item $(\mathrm{ID}_s, \mathrm{ID}_r, U_{\mathrm{ID}_s,t}, X_{\mathrm{ID}_s}, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, R_1, V, t, h_6)$. If it does, C returns h_6. Otherwise, C randomly selects $h_6 \in \{0,1\}^L \times z_q^*$ and repeats the process until h_6 is not in list L_6. C stores the tuple $(\mathrm{ID}_s, \mathrm{ID}_r, U_{\mathrm{ID}_s,t}, X_{\mathrm{ID}_s}, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, R_1, V, t, h_6)$ in list L_6 and returns h_6 to A_{I}.

Partial-Private-Key Query. A_{I} provides a created identity ID.

(1) $\mathrm{ID} \neq \mathrm{ID}_\lambda$. C retrieves y_{ID} from list L_k and returns it to A_{I}.

(2) $\mathrm{ID} = \mathrm{ID}_\lambda$. C aborts.

Secret-Value Query. A_{I} provides a created identity ID. C retrieves x_{ID} from list L_k and returns it to A_{I}.

Public-Key Query. A_{I} provides a created identity ID. C retrieves the public key X_{ID} and public parameters $(Y_{\mathrm{ID}}, T_{\mathrm{ID}})$ from list L_k and returns them to A_{I}.

Public-Key-Replacement Query. A_{I} provides a created identity ID and a new public/secret-value pair $(X'_{\mathrm{ID}}, x'_{\mathrm{ID}})$. C replaces the old public/secret-value pair $(X_{\mathrm{ID}}, x_{\mathrm{ID}})$ with the new one $(X'_{\mathrm{ID}}, x'_{\mathrm{ID}})$ and updates list L_k.

User-Period-Private-Key Query. A_{I} provides a created identity ID and a time period t. C first checks list $L_{\text{Pe-k}}$ to determine whether it contains the item $(\mathrm{ID}, t, u_{\mathrm{ID},t}, U_{\mathrm{ID},t}, s_{\mathrm{ID},t})$. If it does, C returns $(U_{\mathrm{ID},t}, s_{\mathrm{ID},t})$. Otherwise, we consider two cases.

(1) $\mathrm{ID} \neq \mathrm{ID}_\lambda$. C randomly chooses $u_{\mathrm{ID},t} \in Z_q^*$ and computes $U_{\mathrm{ID},t} = u_{\mathrm{ID},t} \cdot P$. C retrieves $(X_{\mathrm{ID}}, Y_{\mathrm{ID}}, T_{\mathrm{ID}})$ from list L_k. Then C makes an H_1 query with the tuple

$(\mathrm{ID}, Y_{\mathrm{ID}}, T_{\mathrm{ID}}, t)$ and obtains a response $h_{1,\mathrm{ID},t}$. C makes an H_2 query with the tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, X_{\mathrm{ID}}, T_{\mathrm{ID}})$ and obtains a response $h_{2,\mathrm{ID}}$. C makes an H_3 query with the tuple $(\mathrm{ID}, Y_{\mathrm{ID}}, U_{\mathrm{ID},t}, t)$ and obtains a response $h_{3,\mathrm{ID},t}$. C retrieves $(x_{\mathrm{ID}}, y_{\mathrm{ID}}, \mathrm{hk}_{\mathrm{ID}})$ from list L_k and computes the period private key $S_{\mathrm{ID},t}$ in time period t as $s_{\mathrm{ID},t} = y_{\mathrm{ID}} + x_{\mathrm{ID}} \cdot h_{2,\mathrm{ID}} + u_{\mathrm{ID},t} \cdot h_{3,\mathrm{ID},t} + \mathrm{hk}_{\mathrm{ID}} \cdot h_{1,\mathrm{ID},t} \bmod q$. C inserts the tuple $(\mathrm{ID}, t, u_{\mathrm{ID},t}, U_{\mathrm{ID},t}, s_{\mathrm{ID},t})$ into list $L_{\text{Pe-k}}$ and returns $(U_{\mathrm{ID},t}, s_{\mathrm{ID},t})$.

(2) $\mathrm{ID} = \mathrm{ID}_\lambda$. C aborts.

GSC Query. A_{I} provides two created identities $\{\mathrm{ID}_s, \mathrm{ID}_r\}$ (one of them may be null), a message m, and a time period t. If ID_s is null, it is equal to an encryption oracle, which only requires the public parameters. Otherwise, we consider two cases.

(1) $\mathrm{ID}_s \neq \mathrm{ID}_\lambda$. C runs the GSC algorithm as normal because C can obtain the private key $s_{\mathrm{ID}_s,t}$ of ID_s in time period t.

(2) $\mathrm{ID}_s = \mathrm{ID}_\lambda$. C first checks list $L_{\text{Pe-k}}$ to determine whether it contains the item $(\mathrm{ID}_\lambda, t, u_{\mathrm{ID}_\lambda,t}, U_{\mathrm{ID}_\lambda,t}, -)$. If it does, C retrieves $(u_{\mathrm{ID}_\lambda,t}, U_{\mathrm{ID}_\lambda,t})$; otherwise, C randomly chooses $u_{\mathrm{ID}_\lambda,t} \in Z_q^*$, computes $U_{\mathrm{ID}_\lambda,t} = u_{\mathrm{ID}_\lambda,t} \cdot P$, and inserts the tuple $(\mathrm{ID}_\lambda, t, u_{\mathrm{ID}_\lambda,t}, U_{\mathrm{ID}_\lambda,t}, -)$ into list $L_{\text{Pe-k}}$.

C randomly chooses $a_1, r_2, h_4 \in Z_q^*$ and computes $R_1 = a_1 \cdot P$, $R_2 = r_2 \cdot P - (Y_{\mathrm{ID}_\lambda} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_\lambda} + X_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + U_{\mathrm{ID}_\lambda,t} \cdot h_{3,\mathrm{ID}_\lambda,t} + T_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t}) \cdot h_4$, $h_5 = H_5(m, R_1, R_2, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r})$, and $u = a_1 \cdot h_5 + r_2 \bmod q$. Then, he/she computes $V = a_1 \cdot (Y_{\mathrm{ID}_r} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_r} + X_{\mathrm{ID}_r} \cdot h_{2,\mathrm{ID}_r} + U_{\mathrm{ID}_r,t} \cdot h_{3,\mathrm{ID}_r,t} + T_{\mathrm{ID}_r} \cdot h_{1,\mathrm{ID}_r,t})$, $h_6 = f(\mathrm{ID}_r) \cdot H_6(\mathrm{ID}_\lambda, \mathrm{ID}_r, U_{\mathrm{ID}_\lambda,t}, X_{\mathrm{ID}_\lambda}, Y_{\mathrm{ID}_\lambda}, T_{\mathrm{ID}_\lambda}, R_1, V, t)$, and $c = m \parallel u \oplus h_6$. The values needed for the above computations can be obtained by querying associated oracles. C inserts the tuple $(m, R_1, R_2, \mathrm{ID}_\lambda, Y_{\mathrm{ID}_\lambda}, \mathrm{ID}_r, U_{\mathrm{ID}_r,t}, X_{\mathrm{ID}_r}, Y_{\mathrm{ID}_r}, T_{\mathrm{ID}_r}, h_4)$ into list L_4. If there is a collision, C rechooses $a_1, r_2, h_4 \in Z_q^*$ and repeats the process. Finally, C returns $\sigma = (R_1, R_2, c)$ to A_{I}.

Un-GSC Query. A_{I} provides two created identities $\{\mathrm{ID}_s, \mathrm{ID}_r\}$ (one of them may be null), a ciphertext σ, and a time period t. If ID_r is null, it is equal to a signature verification oracle, which needs only the public parameters. Otherwise, we consider two cases.

(1) $\mathrm{ID}_r \neq \mathrm{ID}_\lambda$. C runs the Un-GSC algorithm as normal because C can obtain the private key $s_{\mathrm{ID}_r,t}$ of ID_r in time period t.

(2) $\mathrm{ID}_r = \mathrm{ID}_\lambda$. C does not know the period private key of ID_r in time period t. C retrieves $U_{\mathrm{ID}_s,t}$ and $U_{\mathrm{ID}_\lambda,t}$ from list $L_{\text{Pe-k}}$ (if $U_{\mathrm{ID}_s,t}$ or $U_{\mathrm{ID}_\lambda,t}$ does not exist in list $L_{\text{Pe-k}}$, then C makes a period-private-key or GSC query to ensure that they are produced). C starts from the first item of list L_6 to compute $m \parallel u = c \oplus h_6$ and verifies whether the equation $u \cdot P = (Y_{\mathrm{ID}_s} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_s} + X_{\mathrm{ID}_s} \cdot h_{2,\mathrm{ID}_s} + U_{\mathrm{ID}_s,t} \cdot h_{3,\mathrm{ID}_s,t} + T_{\mathrm{ID}_s} \cdot h_{1,\mathrm{ID}_s,t}) \cdot h_4 + R_1 \cdot h_5 + R_2$ holds true, where $h_{0,\mathrm{ID}_s} = H_0(\mathrm{ID}_s, Y_{\mathrm{ID}_s})$, $h_{1,\mathrm{ID}_s,t} = H_1(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, t)$, $h_{2,\mathrm{ID}_s} = H_2(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, X_{\mathrm{ID}_s}, T_{\mathrm{ID}_s})$, $h_{3,\mathrm{ID}_s,t} = H_3(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, U_{\mathrm{ID}_s,t}, t)$, $h_4 = H_4(m, R_1, R_2, \mathrm{ID}_s, Y_{\mathrm{ID}_s}, \mathrm{ID}_\lambda, U_{\mathrm{ID}_\lambda,t}, X_{\mathrm{ID}_\lambda}, Y_{\mathrm{ID}_\lambda}, T_{\mathrm{ID}_\lambda})$, and $h_5 = H_5(m, R_1, R_2, \mathrm{ID}_\lambda, U_{\mathrm{ID}_\lambda,t}, X_{\mathrm{ID}_\lambda}, Y_{\mathrm{ID}_\lambda}, T_{\mathrm{ID}_\lambda})$. The values needed for the above computations can be obtained by querying associated oracles. If the equation holds true, C returns the message m; else C moves to the next item of list L_6 and repeats the process. If no message returns when C traverses all the items of list L_6, C returns $\perp$.

Challenge Stage. A_{I} chooses two different messages (m_0, m_1) with equal length, two created challenge identities $(\mathrm{ID}_s^*, \mathrm{ID}_r^*)$ (ID_s^* may be null), and a time period t^*. If $\mathrm{ID}_r^* \neq \mathrm{ID}_\lambda$, C aborts. Otherwise, C randomly chooses $\eta \in \{0, 1\}$ and randomly chooses $r_2, h_5^* \in Z_q^*$. C retrieves $U_{\mathrm{ID}_s^*,t^*}$ and $U_{\mathrm{ID}_\lambda,t^*}$ from list $L_{\text{Pe-k}}$ (if $U_{\mathrm{ID}_s^*,t^*}$ or $U_{\mathrm{ID}_\lambda,t^*}$ does not exist in list $L_{\text{Pe-k}}$, then C makes a period-private-key or GSC query to ensure that they are produced). C sets $R_1^* = bP$. C randomly chooses $V^* \in G$. C computes $R_2^* = r_2 \cdot P - (bP) \cdot h_5^*$, $h_4^* = H_4(m_\eta, R_1^*, R_2^*, \mathrm{ID}_s^*, Y_{\mathrm{ID}_s^*}, \mathrm{ID}_\lambda, U_{\mathrm{ID}_\lambda,t^*}, X_{\mathrm{ID}_\lambda}, Y_{\mathrm{ID}_\lambda}, T_{\mathrm{ID}_\lambda})$ and $u^* = f(\mathrm{ID}_s^*) \cdot s_{\mathrm{ID}_s^*,t^*} \cdot h_4^* + r_2 \bmod q$. Then, C computes $h_6^* = H_6(\mathrm{ID}_s^*, \mathrm{ID}_\lambda, U_{\mathrm{ID}_s^*,t^*}, X_{\mathrm{ID}_s^*}, Y_{\mathrm{ID}_s^*}, T_{\mathrm{ID}_s^*}, R_1^*, V^*, t^*)$ and $c^* = (m_\eta \parallel u^*) \oplus h_6^*$. The values needed for the above computations can be obtained by querying associated oracles. C inserts the tuple $(m_\eta, R_1^*, R_2^*, \mathrm{ID}_\lambda, U_{\mathrm{ID}_\lambda,t^*}, X_{\mathrm{ID}_\lambda}, Y_{\mathrm{ID}_\lambda}, T_{\mathrm{ID}_\lambda}, h_5^*)$ into list L_5. If there is a collision, C rechooses $r_2, h_5^* \in Z_q^*$ and repeats the process. C returns the challenge ciphertext $\sigma^* = (R_1^*, R_2^*, c^*)$ to A_{I}.

Guess Stage. A_{I} can make the same queries adaptively as in the Find stage with the restrictions given in Definition 1. Finally, A_{I} must give his/her guess η'. A_{I} cannot discover that σ^* is not a valid ciphertext unless he/she asks the H_6 oracle with the tuple $(\mathrm{ID}_s^*, \mathrm{ID}_\lambda, U_{\mathrm{ID}_s^*,t^*}, X_{\mathrm{ID}_s^*}, Y_{\mathrm{ID}_s^*}, T_{\mathrm{ID}_s^*}, R_1^*, V^*, t^*)$, where $V^* = a_1^* \cdot (Y_{\mathrm{ID}_\lambda} + P_{\mathrm{pub}} \cdot h_{0,\mathrm{ID}_\lambda} + X_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + U_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + T_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}) = abP \cdot h_{0,\mathrm{ID}_\lambda} + (r_{\mathrm{ID}_\lambda} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}) \cdot (bP)$. If this occurs, C retrieves r_{ID_λ}, x_{ID_λ}, and $\mathrm{hk}_{\mathrm{ID}_\lambda}$ from list L_k, and it retrieves $u_{\mathrm{ID}_\lambda,t^*}$ and $U_{\mathrm{ID}_\lambda,t^*}$ from list $L_{\text{Pe-k}}$. C makes associated hash oracle queries to obtain $h_{0,\mathrm{ID}_\lambda}$, $h_{1,\mathrm{ID}_\lambda,t^*}$, $h_{2,\mathrm{ID}_\lambda}$, and $h_{3,\mathrm{ID}_\lambda,t^*}$. Then, C can output $abP = [V^* - (r_{\mathrm{ID}_\lambda} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}) \cdot (bP)] \cdot (h_{0,\mathrm{ID}_\lambda})^{-1}$.

Now, we assess the probability of success. In the Challenge stage, the probability that $\mathrm{ID}_r^* = \mathrm{ID}_\lambda$ is $1/q_{\text{U-C}}$. In both the partial-private-key and user-period-private-key queries, the probability of C querying ID_λ is $1/q_{\text{U-C}}$. In the Un-GSC stage, the probability of C refusing the right ciphertext is less than $q_{\text{Un-GSC}}/2^k$.

In terms of the time complexity, GSC and Un-GSC queries need $7t_m$ and $8t_m$ computations, respectively. □

Theorem 11 (type II confidentiality). *In the random oracle model, if there is a PPT adversary A_{II} with a nonnegligible*

advantage ε against the IND-CL-Basic-KI-GSC-CCA2-II security of the scheme running in encryption or signcryption mode in time t and performing at most q_{H_i} H_i ($i = 0, 1, 2, \ldots, 6$) queries, $q_{U\text{-}C}$ user-creation queries, q_{Sec} secret-value queries, $q_{Pe\text{-}p\text{-}k}$ period-private-key queries, q_{GSC} GSC queries, and $q_{Un\text{-}GSC}$ Un-GSC queries, then the CDH problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{U\text{-}C} \cdot (1-1/q_{U\text{-}C})^{q_{Sec}} \cdot (1-1/q_{U\text{-}C})^{q_{Pe\text{-}p\text{-}k}} \cdot (q_{Un\text{-}GSC}/2^k)$ in time $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, where t_m denotes the time for a scalar multiplication on G_1.

Proof. Suppose challenger C is given $(P, aP, bP) \in G_1^3$ for random $a, b \in Z_q^*$. C does not know the values of a and b and is asked to compute abP.

Setup. Adversary A_{II} randomly selects $s \in Z_q^*$ as the master private key and computes the master public key as $P_{\text{pub}} = sP$. Other public parameters are produced normally. A_{II} gives the system public parameters $\text{Params} = \{p, q, E(F_p), G_1, P, P_{\text{pub}}, f, H_0, H_1, H_2, H_3, H_4, H_5, H_6\}$ and master private key s to C. C maintains nine lists $L_0, L_1, L_2, L_3, L_4, L_5, L_6, L_k$, and $L_{\text{Pe-k}}$, which are initially empty. C randomly selects $\lambda \in \{1, 2, \ldots, q_{\text{U-C}}\}$.

Find Stage. A_{II} makes queries to the following oracles adaptively: $H_0, H_1, H_2, H_3, H_4, H_5, H_6$, public-key, user-period-private-key, GSC, and Un-GSC queries are the same as in Theorem 10. The partial-private-key and public-key-replacement queries are not needed for A_{II}.

User-Creation Query. A_{II} provides an identity ID. C looks up list L_k to determine whether it contains the item. If it does, C returns ID's public key X_{ID} and public parameters $(Y_{\text{ID}}, T_{\text{ID}})$ to A_{II}. Otherwise, C proceeds as follows and returns public key X_{ID} and public parameters $(Y_{\text{ID}}, T_{\text{ID}})$ to A_{II}.

C randomly selects $r_{\text{ID}} \in Z_q^*$ and computes $Y_{\text{ID}} = r_{\text{ID}} \cdot P$ and $y_{\text{ID}} = r_{\text{ID}} + s \cdot H_0(\text{ID}, Y_{\text{ID}}) \bmod q$. C randomly selects $\text{hk}_{\text{ID}} \in Z_q^*$ as the helper private key and computes $T_{\text{ID}} = \text{hk}_{\text{ID}} \cdot P$.

(1) $\text{ID} = \text{ID}_\lambda$. C sets the public key as $X_{\text{ID}} = aP$ and inserts the tuple $(\text{ID}_\lambda, X_{\text{ID}_\lambda}, Y_{\text{ID}_\lambda}, -, r_{\text{ID}_\lambda}, y_{\text{ID}_\lambda}, \text{hk}_{\text{ID}_\lambda}, T_{\text{ID}_\lambda})$ into list L_k.

(2) $\text{ID} \neq \text{ID}_\lambda$. C randomly selects $x_{\text{ID}} \in Z_q^*$ as the secret value and computes the public key as $X_{\text{ID}} = x_{\text{ID}} \cdot P$. C inserts the tuple $(\text{ID}, X_{\text{ID}}, Y_{\text{ID}}, x_{\text{ID}}, r_{\text{ID}}, y_{\text{ID}}, \text{hk}_{\text{ID}}, T_{\text{ID}})$ into list L_k.

Secret-Value Query. A_{II} provides a created identity ID.

(1) $\text{ID} \neq \text{ID}_\lambda$. C retrieves x_{ID} from list L_k and returns it to A_{II}

(2) $\text{ID} = \text{ID}_\lambda$. C aborts.

Challenge Stage. The same as in Theorem 10.

Guess Stage. A_{II} can make the same queries adaptively as in the Find stage with the restrictions given in Definition 2. Finally, A_{II} must give his/her guess η'. A_{II} cannot discover that σ^* is not a valid ciphertext unless he/she asks the H_6 oracle with the tuple $(\text{ID}_s^*, \text{ID}_\lambda, U_{\text{ID}_s^*,t^*}, X_{\text{ID}_s^*}, Y_{\text{ID}_s^*}, T_{\text{ID}_s^*}, R_1^*, V^*, t^*)$, where $V^* = a_1^* \cdot (Y_{\text{ID}_\lambda} + P_{\text{pub}} \cdot h_{0,\text{ID}_\lambda} + X_{\text{ID}_\lambda} \cdot h_{2,\text{ID}_\lambda} + U_{\text{ID}_\lambda,t^*} \cdot h_{3,\text{ID}_\lambda,t^*} + T_{\text{ID}_\lambda} \cdot h_{1,\text{ID}_\lambda,t^*}) = abP \cdot h_{2,\text{ID}_\lambda} + (r_{\text{ID}_\lambda} + s \cdot h_{0,\text{ID}_\lambda} + u_{\text{ID}_\lambda,t^*} \cdot h_{3,\text{ID}_\lambda,t^*} + \text{hk}_{\text{ID}_\lambda} \cdot h_{1,\text{ID}_\lambda,t^*}) \cdot (bP)$. If this occurs, C retrieves r_{ID_λ} and $\text{hk}_{\text{ID}_\lambda}$ from list L_k as well as $u_{\text{ID}_\lambda,t^*}$ and $U_{\text{ID}_\lambda,t^*}$ from list $L_{\text{Pe-k}}$. C makes associated hash oracles to obtain $h_{0,\text{ID}_\lambda}, h_{1,\text{ID}_\lambda,t^*}, h_{2,\text{ID}_\lambda}$, and $h_{3,\text{ID}_\lambda,t^*}$. Then, C can output $abP = [V^* - (r_{\text{ID}_\lambda} + s \cdot h_{0,\text{ID}_\lambda} + u_{\text{ID}_\lambda,t^*} \cdot h_{3,\text{ID}_\lambda,t^*} + \text{hk}_{\text{ID}_\lambda} \cdot h_{1,\text{ID}_\lambda,t^*}) \cdot bP] \cdot (h_{2,\text{ID}_\lambda})^{-1}$.

Now, we assess the probability of success. In the challenge stage, the probability that $\text{ID}_r^* = \text{ID}_\lambda$ is $1/q_{\text{U-C}}$. In both the secret-value and user-period-private-key queries, the probability of C querying ID_λ is $1/q_{\text{U-C}}$. In the Un-GSC stage, the probability of C refusing the right ciphertext is less than $q_{\text{Un-GSC}}/2^k$.

In terms of the time complexity, GSC and Un-GSC queries need $7t_m$ and $8t_m$ computations, respectively. □

6.2. Unforgeability of Basic Key Insulation

Theorem 12 (type I unforgeability). *In the random oracle model, if there is a PPT adversary A_I with a nonnegligible advantage ε against the EUF-CL-Basic-KI-GSC-CMA-I security of the scheme running in signature or signcryption mode in time t and performing at most q_{H_i} H_i ($i = 0, 1, 2, \ldots, 6$) queries, $q_{U\text{-}C}$ user-creation queries, $q_{Pa\text{-}p\text{-}k}$ partial-private-key queries, $q_{Pe\text{-}p\text{-}k}$ period-private-key queries, q_{GSC} GSC queries, and $q_{Un\text{-}GSC}$ Un-GSC queries, then the EC-DL problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{U\text{-}C} \cdot (1-1/q_{U\text{-}C})^{q_{Pa\text{-}p\text{-}k}} \cdot (1-1/q_{U\text{-}C})^{q_{Pe\text{-}p\text{-}k}} \cdot (q_{Un\text{-}GSC}/2^k) \cdot (\varepsilon^3/(q_{H_0} + q_{H_4})^6 - 3/2^k)$ in time $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, where t_m denotes the time for a scalar multiplication on G_1.*

Proof. Suppose that challenger C is given $(P, aP) \in G_1^2$ for random $a \in Z_q^*$. C does not know the value of a and is asked to compute a. To utilize adversary A_{I}, challenger C will simulate all the oracles defined in Definition 3.

Setup. The same as in Theorem 10.

Queries. The same as in Theorem 10.

Forgery. Finally, A_{I} outputs a forged GSC ciphertext $\sigma^* = (R_1^*, R_2^*, c^*)$ in message m^* in time period t^* with ID_s^* as the sender and ID_r^* as the receiver. If $\text{ID}_s^* \neq \text{ID}_\lambda$, C aborts. Otherwise, if σ^* can pass the validation of the Un-GSC algorithm and A_{I} does not violate the restrictions of Definition 3, according to the multiple-forking lemma [45], we can obtain four valid signatures: $(\text{ID}_\lambda, \text{ID}_r^*, m^*, u^{*(1)}, h_{0,\text{ID}_\lambda}^{(1)}, h_4^{*(1)})$, $(\text{ID}_\lambda, \text{ID}_r^*, m^*, u^{*(2)}, h_{0,\text{ID}_\lambda}^{(1)}, h_4^{*(2)})$, $(\text{ID}_\lambda, \text{ID}_r^*, m^*, u^{*(3)}, h_{0,\text{ID}_\lambda}^{(2)}, h_4^{*(3)})$, and $(\text{ID}_\lambda, \text{ID}_r^*, m^*, u^{*(4)}, h_{0,\text{ID}_\lambda}^{(2)}, h_4^{*(4)})$, where $h_{0,\text{ID}_\lambda}^{(1)}$ and $h_{0,\text{ID}_\lambda}^{(2)}$ are two different hash values corresponding to the H_0 oracle and $h_4^{*(1)}$, $h_4^{*(2)}$, $h_4^{*(3)}$, and $h_4^{*(4)}$ are four different hash values corresponding to the H_4 oracle. Because

$u^* = s_{\mathrm{ID}_\lambda,t^*} \cdot h_4^* + a_1^* \cdot h_5^* + a_2^* \bmod q$, we can obtain four equations:

$$\begin{aligned}
u^{*(1)} &= \left(r_{\mathrm{ID}_\lambda} + a \cdot h_{0,\mathrm{ID}_\lambda}^{(1)} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(1)} + a_1^* \cdot h_5^* + a_2^* \bmod q,\\
u^{*(2)} &= \left(r_{\mathrm{ID}_\lambda} + a \cdot h_{0,\mathrm{ID}_\lambda}^{(1)} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(2)} + a_1^* \cdot h_5^* + a_2^* \bmod q,\\
u^{*(3)} &= \left(r_{\mathrm{ID}_\lambda} + a \cdot h_{0,\mathrm{ID}_\lambda}^{(2)} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(3)} + a_1^* \cdot h_5^* + a_2^* \bmod q,\\
u^{*(4)} &= \left(r_{\mathrm{ID}_\lambda} + a \cdot h_{0,\mathrm{ID}_\lambda}^{(2)} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(4)} + a_1^* \cdot h_5^* + a_2^* \bmod q.
\end{aligned} \tag{3}$$

Then, C can compute

$$\begin{aligned}
&\left(u^{*(1)} - u^{*(2)}\right)\left(h_4^{*(1)} - h_4^{*(2)}\right)^{-1} = r_{\mathrm{ID}_\lambda} + a \cdot h_{0,\mathrm{ID}_\lambda}^{(1)} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*},\\
&\left(u^{*(3)} - u^{*(4)}\right)\left(h_4^{*(3)} - h_4^{*(4)}\right)^{-1} = r_{\mathrm{ID}_\lambda} + a \cdot h_{0,\mathrm{ID}_\lambda}^{(2)} + x_{\mathrm{ID}_\lambda} \cdot h_{2,\mathrm{ID}_\lambda} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*},\\
&\left[\left(u^{*(1)} - u^{*(2)}\right)\left(h_4^{*(1)} - h_4^{*(2)}\right)^{-1} - \left(u^{*(3)} - u^{*(4)}\right)\left(h_4^{*(3)} - h_4^{*(4)}\right)^{-1}\right] \cdot \left(h_{0,\mathrm{ID}_\lambda}^{(1)} - h_{0,\mathrm{ID}_\lambda}^{(2)}\right)^{-1} = a.
\end{aligned} \tag{4}$$

Now, we assess the probability of success. In the Forgery stage, the probability of $\mathrm{ID}_s^* = \mathrm{ID}_\lambda$ is $1/q_{\text{U-C}}$. In both the partial-private-key and user-period-private-key queries, the probability of C querying with ID_λ is $1/q_{\text{U-C}}$. In the Un-GSC stage, the probability of C refusing the right ciphertext is less than $q_{\text{Un-GSC}}/2^k$. In conjunction with the multiple-forking lemma, the EC-DL problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{\text{U-C}} \cdot (1-1/q_{\text{U-C}})^{q_{\text{Pa-p-k}}} \cdot (1-1/q_{\text{U-C}})^{q_{\text{Pe-p-k}}} \cdot (q_{\text{Un-GSC}}/2^k) \cdot (\varepsilon^3/(q_{H_0} + q_{H_4})^6 - 3/2^k)$.

In terms of the time complexity, GSC and Un-GSC queries need $7t_m$ and $8t_m$ computations, respectively. □

Theorem 13 (type II unforgeability). *In the random oracle model, if there is a PPT adversary A_{II} with a nonnegligible advantage ε against the EUF-CL-Basic-KI-GSC-CMA-II security of the scheme running in signature or signcryption mode in time t and performing at most q_{H_i} H_i $(i = 0, 1, 2, \ldots, 6)$ queries, $q_{\text{U-C}}$ user-creation queries, q_{Sec} secret-value queries, $q_{Pe\text{-}p\text{-}k}$ period-private-key queries, q_{GSC} GSC queries, and $q_{Un\text{-}GSC}$ Un-GSC queries, then the EC-DL problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{\text{U-C}} \cdot (1 - 1/q_{\text{U-C}})^{q_{Secret}} \cdot (1 - 1/q_{\text{U-C}})^{q_{Pe\text{-}p\text{-}k}} \cdot (q_{Un\text{-}GSC}/2^k) \cdot (\varepsilon^3/(q_{H_2} + q_{H_4})^6 - 3/2^k)$ in time $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, where t_m denotes the time for a scalar multiplication on G_1.*

Proof. Suppose that challenger C is given $(P, aP) \in G_1^2$ for random $a \in Z_q^*$. C does not know the value of a and is asked to compute a. To utilize adversary A_{II}, challenger C will simulate all the oracles defined in Definition 4.

Setup. The same as in Theorem 11.

Queries. The same as in Theorem 11.

Forgery. Finally, A_{II} outputs a forged GSC ciphertext $\sigma^* = (R_1^*, R_2^*, c^*)$ in message m^* in time period t^* with ID_s^* as the sender and ID_r^* as the receiver. If $\mathrm{ID}_s^* \neq \mathrm{ID}_\lambda$, C aborts. Otherwise, if σ^* can pass the validation of the Un-GSC algorithm and A_{II} does not violate the restrictions of Definition 4, according to the multiple-forking lemma [45], we can obtain four valid signatures: $(\mathrm{ID}_\lambda, \mathrm{ID}_r^*, m^*, u^{*(1)}, h_{2,\mathrm{ID}_\lambda}^{(1)}, h_4^{*(1)})$, $(\mathrm{ID}_\lambda, \mathrm{ID}_r^*, m^*, u^{*(2)}, h_{2,\mathrm{ID}_\lambda}^{(1)}, h_4^{*(2)})$, $(\mathrm{ID}_\lambda, \mathrm{ID}_r^*, m^*, u^{*(3)}, h_{2,\mathrm{ID}_\lambda}^{(2)}, h_4^{*(3)})$, and $(\mathrm{ID}_\lambda, \mathrm{ID}_r^*, m^*, u^{*(4)}, h_{2,\mathrm{ID}_\lambda}^{(2)}, h_4^{*(4)})$, where $h_{2,\mathrm{ID}_\lambda}^{(1)}$ and $h_{2,\mathrm{ID}_\lambda}^{(2)}$ are two different hash values corresponding to the H_2 oracle and $h_4^{*(1)}$, $h_4^{*(2)}$, $h_4^{*(3)}$, and $h_4^{*(4)}$ are four different hash values corresponding to the H_4 oracle. Because $u^* = s_{\mathrm{ID}_\lambda,t^*} \cdot h_4^* + a_1^* \cdot h_5^* + a_2^* \bmod q$, we can obtain four equations:

$$\begin{aligned}
u^{*(1)} &= \left(r_{\mathrm{ID}_\lambda} + s \cdot h_{0,\mathrm{ID}_\lambda} + a \cdot h_{2,\mathrm{ID}_\lambda}^{(1)} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(1)} + a_1^* \cdot h_5^* + a_2^* \bmod q,\\
u^{*(2)} &= \left(r_{\mathrm{ID}_\lambda} + s \cdot h_{0,\mathrm{ID}_\lambda} + a \cdot h_{2,\mathrm{ID}_\lambda}^{(1)} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(2)} + a_1^* \cdot h_5^* + a_2^* \bmod q,\\
u^{*(3)} &= \left(r_{\mathrm{ID}_\lambda} + s \cdot h_{0,\mathrm{ID}_\lambda} + a \cdot h_{2,\mathrm{ID}_\lambda}^{(2)} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(3)} + a_1^* \cdot h_5^* + a_2^* \bmod q,\\
u^{*(4)} &= \left(r_{\mathrm{ID}_\lambda} + s \cdot h_{0,\mathrm{ID}_\lambda} + a \cdot h_{2,\mathrm{ID}_\lambda}^{(2)} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*}\right) \cdot h_4^{*(4)} + a_1^* \cdot h_5^* + a_2^* \bmod q.
\end{aligned} \tag{5}$$

Then, C can compute

$$\left(u^{*(1)} - u^{*(2)}\right)\left(h_4^{*(1)} - h_4^{*(2)}\right)^{-1} = r_{\mathrm{ID}_\lambda} + s \cdot h_{0,\mathrm{ID}_\lambda} + a \cdot h_{2,\mathrm{ID}_\lambda}^{(1)} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*},$$

$$\left(u^{*(3)} - u^{*(4)}\right)\left(h_4^{*(3)} - h_4^{*(4)}\right)^{-1} = r_{\mathrm{ID}_\lambda} + s \cdot h_{0,\mathrm{ID}_\lambda} + a \cdot h_{2,\mathrm{ID}_\lambda}^{(2)} + u_{\mathrm{ID}_\lambda,t^*} \cdot h_{3,\mathrm{ID}_\lambda,t^*} + \mathrm{hk}_{\mathrm{ID}_\lambda} \cdot h_{1,\mathrm{ID}_\lambda,t^*},$$
$$\left[\left(u^{*(1)} - u^{*(2)}\right)\left(h_4^{*(1)} - h_4^{*(2)}\right)^{-1} - \left(u^{*(3)} - u^{*(4)}\right)\left(h_4^{*(3)} - h_4^{*(4)}\right)^{-1}\right] \cdot \left(h_{2,\mathrm{ID}_\lambda}^{(1)} - h_{2,\mathrm{ID}_\lambda}^{(2)}\right)^{-1} = a. \tag{6}$$

Now, we assess the probability of success. In the Challenge stage, the probability of $\mathrm{ID}_s^* = \mathrm{ID}_\lambda$ is $1/q_{\text{U-C}}$. In both the secret-value and user-period-private-key queries, the probability of C querying ID_λ is $1/q_{\text{U-C}}$. In the Un-GSC stage, the probability of C refusing the right ciphertext is less than $q_{\text{Un-GSC}}/2^k$. In conjunction with the multiple-forking lemma, the EC-GDL problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{\text{U-C}} \cdot (1 - 1/q_{\text{U-C}})^{q_{\text{Secret}}} \cdot (1 - 1/q_{\text{U-C}})^{q_{\text{Pe-p-k}}} \cdot (q_{\text{Un-GSC}}/2^k) \cdot (\varepsilon^3/(q_{H_2} + q_{H_4})^6 - 3/2^k)$.

In terms of the time complexity, GSC and Un-GSC queries need $7t_m$ and $8t_m$ computations, respectively. □

6.3. Confidentiality of Strong Key Insulation

Theorem 14 (type I confidentiality). *In the random oracle model, if there is a PPT adversary A_I with a nonnegligible advantage ε against the IND-CL-Strong-KI-GSC-CCA2-I security of the scheme running in encryption or signcryption mode in time t and performing at most q_{H_i} H_i $(i = 0, 1, 2, \ldots, 6)$ queries, $q_{U\text{-}C}$ user-creation queries, $q_{Pa\text{-}p\text{-}k}$ partial-private-key queries, q_{GSC} GSC queries, and $q_{Un\text{-}GSC}$ Un-GSC queries, then the CDH problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{U\text{-}C} \cdot (1 - 1/q_{U\text{-}C})^{q_{Pa\text{-}p\text{-}k}} \cdot (q_{Un\text{-}GSC}/2^k)$ in time $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, where t_m denotes the time for a scalar multiplication on G_1.*

Proof. The proof is almost the same as that of Theorem 10. The difference is that A_{I} cannot make a user-period-private-key query but can adaptively ask the helper-key oracle. The GSC oracle is also slightly different.

Helper-Key Query. A_{I} provides a created identity ID and a time period t. C returns the user's helper key $\mathrm{hk}_{\mathrm{ID}}$ and the ephemeral variable $u_{\mathrm{ID},t}$ in time period t to A_{I}.

GSC Query. A_{I} provides two created identities $\{\mathrm{ID}_s, \mathrm{ID}_r\}$ (one of them may be null), a message m, and a time period t. If ID_s is null, it is equal to an encryption oracle, which needs only the public parameters. Otherwise, we consider two cases.

(1) $\mathrm{ID}_s \neq \mathrm{ID}_\lambda$. C first checks list $L_{\text{Pe-k}}$ to determine whether it contains the item $(\mathrm{ID}_s, t, u_{\mathrm{ID}_s,t}, U_{\mathrm{ID}_s,t}, s_{\mathrm{ID}_s,t})$. If it does, C retrieves the period private key $s_{\mathrm{ID}_s,t}$ in time period t and runs the GSC algorithm as normal. Otherwise, C randomly chooses $u_{\mathrm{ID}_s,t} \in Z_q^*$ and computes $U_{\mathrm{ID}_s,t} = u_{\mathrm{ID}_s,t} \cdot P$. C retrieves $(X_{\mathrm{ID}_s}, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s})$ from list L_k. Then, C makes an H_1 query with the tuple $(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, T_{\mathrm{ID}_s}, t)$ and obtains a response $h_{1,\mathrm{ID}_s,t}$. C makes an H_2 query with the tuple $(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, X_{\mathrm{ID}_s}, T_{\mathrm{ID}_s})$ and obtains a response h_{2,ID_s}. C makes an H_3 query with the tuple $(\mathrm{ID}_s, Y_{\mathrm{ID}_s}, U_{\mathrm{ID}_s,t}, t)$ and obtains a response $h_{3,\mathrm{ID}_s,t}$. C retrieves $(x_{\mathrm{ID}_s}, y_{\mathrm{ID}_s}, \mathrm{hk}_{\mathrm{ID}_s})$ from list L_k and computes the period private key $S_{\mathrm{ID}_s,t}$ in time period t as $s_{\mathrm{ID}_s,t} = y_{\mathrm{ID}_s} + x_{\mathrm{ID}_s} \cdot h_{2,\mathrm{ID}_s} + u_{\mathrm{ID}_s,t} \cdot h_{3,\mathrm{ID}_s,t} + \mathrm{hk}_{\mathrm{ID}_s} \cdot h_{1,\mathrm{ID}_s,t} \bmod q$. C inserts the tuple $(\mathrm{ID}_s, t, u_{\mathrm{ID}_s,t}, U_{\mathrm{ID}_s,t}, s_{\mathrm{ID}_s,t})$ into list $L_{\text{Pe-k}}$ and runs the GSC algorithm as normal.

(2) $\mathrm{ID}_s = \mathrm{ID}_\lambda$. It is the same as in Theorem 10.

Finally, C returns the GSC ciphertext $\sigma = (R_1, R_2, c)$ and $U_{\mathrm{ID}_s,t}$ to A_{I}. □

Theorem 15 (type II confidentiality). *In the random oracle model, if there is a PPT adversary A_{II} with a nonnegligible advantage ε against the IND-CL-Strong-KI-GSC-CCA2-II security of the scheme running in encryption or signcryption mode in time t and performing at most q_{H_i} H_i $(i = 0, 1, 2, \ldots, 6)$ queries, $q_{U\text{-}C}$ user-creation queries, q_{Sec} secret-value queries, q_{GSC} GSC queries, and $q_{Un\text{-}GSC}$ Un-GSC queries, then the CDH problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{U\text{-}C} \cdot (1 - 1/q_{U\text{-}C})^{q_{Sec}} \cdot (q_{Un\text{-}GSC}/2^k)$ in time $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, where t_m denotes the time for a scalar multiplication on G_1.*

Proof. The proof is almost the same as that of Theorem 11. The difference is that A_{II} cannot make a user-period-private-key query but can adaptively ask the helper-key oracle.

The user-creation query, secret-value query, public-key query, and Un-GSC queries are the same as in Theorem 11.

The helper-key query and GSC query are the same as in Theorem 14. □

6.4. Unforgeability of Strong Key Insulation

Theorem 16 (type I unforgeability). *In the random oracle model, if there is a PPT adversary A_I with a nonnegligible advantage ε against the EUF-CL-Strong-KI-GSC-CMA-I security of the scheme running in signature or signcryption mode in time t and performing at most q_{H_i} H_i $(i = 0, 1, 2, \ldots, 6)$ queries, $q_{U\text{-}C}$ user-creation queries, $q_{Pa\text{-}p\text{-}k}$ partial-private-key queries, q_{GSC} GSC queries, and $q_{Un\text{-}GSC}$ Un-GSC queries, then the EC-DL problem can be solved with probability $\varepsilon' \geq \varepsilon \cdot 1/q_{U\text{-}C} \cdot (1 - 1/q_{U\text{-}C})^{q_{Pa\text{-}p\text{-}k}} \cdot (q_{Un\text{-}GSC}/2^k) \cdot (\varepsilon^3/(q_{H_0} + q_{H_4})^6 - 3/2^k)$ in time $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, where t_m denotes the time for a scalar multiplication on G_1.*

Proof. The proof is almost the same as that of Theorem 12. The difference is that A_{I} cannot make a user-period-private-key query but can adaptively ask the helper-key oracle.

All the oracle queries are the same as in Theorem 14. □

Theorem 17 (type II unforgeability). *In the random oracle model, if there is a PPT adversary A_{II} with a nonnegligible advantage ε against the EUF-CL-Strong-KI-GSC-CMA-II security of the scheme running in signature or signcryption mode in time t and performing at most*

Table 1: Comparison of computational costs.

Schemes	S-I-K	K-U-H	K-U-U	Sc	Un-Sc	Model	Technology
[33]	$4m_1$	$3m_1$	0	$6m_1 + e$	$7p$	S	Identity-based
[34]	$5m_1$	$4m_1$	0	$p + 5m_1 + e$	$6p + 2m_1$	S	PKI-based
[35]	m_1	m_1	0	$p + 2m_1 + e$	$5p + 6m_1$	ROM	Identity-based
[38]	$4m_1$	$3m_1$	0	$6m_1 + e$	$7p$	S	Identity-based
[43]	0	0	0	$(4n + 4)m_2$	$(4n + 4)m_2$	ROM	Attribute-based
Ours	m_2	m_2	0	$7m_2$	$8m_2$	ROM	Certificateless

Note. n represents the number of attributes.

q_{H_i} H_i $(i = 0, 1, 2, \ldots, 6)$ *queries,* $q_{U\text{-}C}$ *user-creation queries,* q_{Sec} *secret-value queries,* q_{GSC} *GSC queries, and* $q_{Un\text{-}GSC}$ *Un-GSC queries, then the EC-DL problem can be solved with probability* $\varepsilon' \geq \varepsilon \cdot 1/q_{U\text{-}C} \cdot (1 - 1/q_{U\text{-}C})^{q_{Secret}} \cdot (q_{Un\text{-}GSC}/2^k) \cdot (\varepsilon^3/(q_{H_2} + q_{H_4})^6 - 3/2^k)$ *in time* $t' < t + (7 \cdot q_{GSC} + 8 \cdot q_{Un\text{-}GSC}) \cdot t_m$, *where* t_m *denotes the time for a scalar multiplication on* G_1.

Proof. The proof is almost the same as that of Theorem 13. The difference is that A_{II} cannot make a user-period-private-key query but can adaptively ask the helper-key oracle.

All the oracle queries are the same as in Theorem 15. □

6.5. Secure Key Update

Theorem 18. *Our certificateless key-insulated GSC scheme supports secure key update.*

Proof. If an adversary compromises the user's storage while a key is being updated from time period t to t', then he/she can obtain the user's period private key $s_{\text{ID},t}$ in time period t and the update key $\text{uk}_{\text{ID},t,t'}$ from t to t'. Then, he/she can compute the user's period private key $s_{\text{ID},t'} = s_{\text{ID},t} + \text{uk}_{\text{ID},t,t'}$ in time period t'. The adversary cannot obtain any other useful information except $s_{\text{ID},t}$, $\text{uk}_{\text{ID},t,t'}$, and $s_{\text{ID},t'}$. In addition, from the equation $s_{\text{ID},t} = y_{\text{ID}} + x_{\text{ID}} \cdot h_{2,\text{ID}} + u_{\text{ID},t} \cdot h_{3,\text{ID},t} + \text{hk}_{\text{ID}} \cdot h_{1,\text{ID},t} \bmod q$, he/she cannot derive any private information because the number of unknown variables is greater than the number of equations. Furthermore, he/she cannot derive any private information from the equation $\text{uk}_{\text{ID},t,t'} = u_{\text{ID},t'} \cdot h_{3,\text{ID},t'} - u_{\text{ID},t} \cdot h_{3,\text{ID},t} + \text{hk}_{\text{ID}} \cdot (h_{1,\text{ID},t'} - h_{1,\text{ID},t}) \bmod q$ either. □

7. Efficiency Comparison of the Proposed Scheme

Because a certificateless key-insulated signcryption scheme has yet to be proposed and because only five key-insulated signcryption schemes have been reported in the literature, we compare the performance of our scheme in signcryption mode with these schemes. These five schemes are Chen et al.'s identity-based key-insulated signcryption scheme [33], Fan et al.'s PKI-based key-insulated signcryption scheme [34], Wang et al.'s identity-based key-insulated signcryption scheme [35], Zhu et al.'s identity-based key-insulated signcryption scheme [38], and Hong and Sun's attribute-based key-insulated signcryption scheme [43]. Their common computations are set-initial-key, key-update-h, key-update-u, signcryption, and un-signcryption. The comparisons are presented in Tables 1 and 2. The symbols p, m_1, m_2, and e denote a pairing computation, a pairing-based scalar multiplication computation on G_1, an ECC-based scalar multiplication computation on G_1, and an exponentiation computation on G_2, respectively. Other computations are ignored because they are not time consuming. $|G_1|$, $|G_2|$, $|q|$, $|m|$, and $|\text{ID}|$ represent the bit length of an element on G_1, G_2, and z_q^*, a message m, and an identity, respectively. "S" denotes the standard model. "ROM" denotes the random oracle model. S-I-K, K-U-H, K-U-U, Sc, and Un-Sc denote set-initial-key, key-update-h, key-update-u, signcryption, and un-signcryption, respectively.

Table 2: Comparison of communication overheads.

Schemes	Ciphertext size
[33]	$6\|G_1\| + \|G_2\|$
[34]	$4\|G_1\| + \|G_2\| + \|q\|$
[35]	$2\|G_1\| + \|G_2\| + \|q\|$
[38]	$6\|G_1\| + \|G_2\|$
[43]	$5\|G_1\| + 2\|q\| + \|m\|$
Ours	$2\|G_1\| + \|q\| + \|m\|$

Table 3: Computational costs (milliseconds).

p	m_1	m_2	e
14.90	4.31	0.97	1.25

To show the comparisons more directly, we use the Multiprecision Integer and Rational Arithmetic C Library (MIRACL) [46] to test the runtime of the basic cryptographic operations. The average runtime is shown in Table 3 (we tested it 1000 times). The experiment was run on a Windows 7 Home Basic 64-bit operating system. The hardware consisted of an Intel Core i7-4510U CPU running at 2.0 GHz with 8 GB of memory. For pairing-based schemes, we use the supersingular elliptic curve $E/F_p : y^2 = x^3 - 3x$ with an embedding degree of 2, where q is a 160-bit Solinas prime $q = 2^{159} + 2^{17} + 1$ and p is a 512-bit prime satisfying $p + 1 = 2qh$. Its security level is equivalent to 80-bit Advanced Encryption Standard (AES). To achieve the same security level, for elliptic curve cryptography- (ECC-) based schemes, we use secp160r1, which is recommended by Certicom Corporation [47].

When we take the above parameters, for pairing-based schemes, $|G_1| = |G_2| = 1024$ and $|q| = 160$; for ECC-based

Table 4: Comparison of performances ($n = 2$).

Schemes	S-I-K	K-U-H	K-U-U	Sc	Un-Sc	Ciphertext
[33]	17.24	12.93	0	27.11	104.30	7168
[34]	21.55	17.24	0	37.70	98.02	5280
[35]	4.31	4.31	0	24.77	100.36	3232
[38]	17.24	12.93	0	27.11	104.30	7168
[43]	0	0	0	11.64	11.64	2080
Ours	0.97	0.97	0	6.79	7.76	960

Table 5: Comparison of performances.

Schemes	Sc	Un-Sc	Model	Ciphertext size
[13]	$4m_1 + e$	$4p + m_1$	ROM	$2\lvert G_1\rvert + \lvert m\rvert$
[20]	$4m_2$	$5m_2$	ROM	$\lvert G_1\rvert + 2\lvert q\rvert + \lvert m\rvert$
Ours	$7m_2$	$8m_2$	ROM	$2\lvert G_1\rvert + \lvert q\rvert + \lvert m\rvert$

Table 6: Comparison of performances.

Schemes	Sc	Un-Sc	Model	Ciphertext size
[13]	18.49	63.91	ROM	2208
[20]	3.88	4.85	ROM	800
Ours	6.79	7.76	ROM	960

schemes, $|G_1| = 320$ and $|q| = 160$. Let $|m| = 160$. We can obtain Table 4 by combining Tables 1, 2, and 3.

From Table 4, we can see that scheme [43] is the most efficient one in the S-I-K and K-U-H stages. Our scheme is very similar to scheme [43] and outperforms other schemes. In the Sc and Un-Sc stages, our scheme is the most efficient scheme. Scheme [43] is very similar to our scheme, whereas other schemes are much less efficient than our scheme. In terms of ciphertext size, our scheme is the shortest.

We also compared our scheme with two certificateless GSC schemes. These schemes are Zhou et al.'s scheme [13] and Zhang et al.'s scheme [20]. The comparisons are shown in Table 5. Let $|\mathrm{ID}| = 160$, and we can obtain Table 6 by combining Tables 3 and 5.

From Table 6, we can see that scheme [20] is the most efficient scheme in terms of Sc, Un-Sc, and ciphertext size. Scheme [20] is a lightweight scheme and achieves the greatest efficiency. Our scheme is very similar to this scheme and outperforms scheme [13]. None of the schemes [13, 20] consider the private key exposure problem, whereas our scheme achieves high efficiency even after considering this problem.

In general, the efficiency of our scheme is very similar to those of the lightweight schemes [20, 43]; therefore, our scheme is more suitable for users who communicate with the cloud using mobile devices.

8. Conclusions

In this paper, we propose a certificateless key-insulated GSC scheme without bilinear pairings. Our scheme can be used to ensure cloud storage security. We provide a formal definition and security model of certificateless key-insulated GSC. Our scheme is demonstrated to be confidential under the CDH assumption and unforgeable under the EC-DL assumption, and it supports both random-access key update and secure key update. Efficiency evaluations show that our scheme is efficient compared with current key-insulated signcryption schemes and certificateless GSC schemes. Our future work will include designing highly efficient intrusion-resilient GSC schemes.

Conflicts of Interest

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

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant nos. 61462048, 61562047, and 61662039. The authors express their thanks to Ms. Yan Di, who checked their manuscript.

References

[1] Y. L. Zheng, "Digital signcryption or how to achieve cost (signature & encryption) << cost (signature) + cost(encryption)," in *Advances in cryptology-CRYPTO 1997*, pp. 165–179, Springer, Berlin, Germany, 1997.

[2] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in cryptology- CRYPTO 1984*, Lecture Notes in Computer Science, pp. 47–53, Springer, Berlin, Germany, 1984.

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

[4] Y. Han, X. Yang, P. Wei, Y. Wang, and Y. Hu, "ECGSC: Elliptic Curve Based Generalized Signcryption," in *Ubiquitous Intelligence and Computing, Third International Conference*, vol. 4159 of *Lecture Notes in Computer Science*, pp. 956–965, Springer, Berlin, Germany, 2006.

[5] R. Anderson, "Two remarks on public key cryptology," in *Proceedings of the 4th ACM Conference on Computer and Communications Security*, ACM, 1997.

[6] Y. Dodis, J. Katz, S. Xu, and M. Yung, "Key-insulated public key cryptosystems," in *Advances in cryptology- EUROCRYPT 2002*, vol. 2332 of *Lecture Notes in Comput. Sci.*, pp. 65–82, Springer, Berlin, Germany, 2002.

[7] G. Itkis and L. Reyzin, "SiBIR: Signer-base intrusion-resilient signatures," in *Advances in cryptology—CRYPTO 2002*, vol. 2442 of *Lecture Notes in Comput. Sci.*, pp. 499–514, Springer, Berlin, Germany, 2002.

[8] L. Chen, Z. Cheng, and N. P. Smart, "Identity-based key agreement protocols from pairings," *International Journal of Information Security*, vol. 6, no. 4, pp. 213–241, 2007.

[9] Y. Han and X. Gui, "Adaptive secure multicast in wireless networks," *International Journal of Communication Systems*, vol. 22, no. 9, pp. 1213–1239, 2009.

[10] X. A. Wang, X. Yang, and J. Zhang, "Provable secure generalized signcryption," *Journal of Computers*, vol. 5, no. 5, pp. 807–814, 2010.

[11] G. Yu, X. Ma, Y. Shen, and W. Han, "Provable secure identity based generalized signcryption scheme," *Theoretical Computer Science*, vol. 411, no. 40–42, pp. 3614–3624, 2010.

[12] P. Kushwah and S. Lal, "An efficient identity based generalized signcryption scheme," *Theoretical Computer Science*, vol. 412, no. 45, pp. 6382–6389, 2011.

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

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

[15] G. Wei, J. Shao, Y. Xiang, P. Zhu, and R. Lu, "Obtain confidentiality or/and authenticity in big data by ID-based generalized signcryption," *Information Sciences. An International Journal*, vol. 318, pp. 111–122, 2015.

[16] C.-X. Zhou, "An improved multi-receiver generalized signcryption scheme," *International Journal of Network Security*, vol. 17, no. 3, pp. 340–350, 2015.

[17] Y. Han and W. Lu, "Attribute based generalized signcryption for online social network," in *Proceedings of the 34th Chinese Control Conference*, pp. 6434–6439, July 2015.

[18] C.-X. Zhou, "Identity based generalized proxy signcryption scheme," *Information Technology and Control*, vol. 45, no. 1, pp. 13–26, 2016.

[19] C. Zhou, Z. Cui, and G. Gao, "Efficient identity-based generalized ring signcryption scheme," *KSII Transactions on Internet and Information Systems*, vol. 10, no. 12, pp. 5553–5571, 2016.

[20] A. Zhang, L. Wang, X. Ye, and X. Lin, "Light-Weight and Robust Security-Aware D2D-Assist Data Transmission Protocol for Mobile-Health Systems," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 3, pp. 662–675, 2017.

[21] Y. Dodis, J. Katz, S. Xu, and M. Yung, "Strong key-insulated signature schemes," in *Public Key Cryptography—PKC 2003*, vol. 2567 of *Lecture Notes in Computer Science*, pp. 130–144, Springer, Berlin, Germany, 2002.

[22] Y. Hanaoka, G. Hanaoka, J. Shikata, and H. Imai, "Identity-based hierarchical strongly key-insulated encryption and its application," in *Advances in cryptology—ASIACRYPT 2005*, vol. 3788 of *Lecture Notes in Comput. Sci.*, pp. 495–514, Springer, Berlin, Germany, 2005.

[23] Y. Zhou, Z. Cao, and Z. Chai, "Identity Based Key Insulated Signature," in *Information Security Practice and Experience*, vol. 3903 of *Lecture Notes in Computer Science*, pp. 226–234, Springer, Berlin, Germany, 2006.

[24] J. Weng, K. Chen, S. Liu, and C. Ma, "Identity-based key-insulated signature without random oracles," in *Proceedings of the 2006 International Conference on Computational Intelligence and Security, ICCIAS 2006*, pp. 1253–1258, October 2006.

[25] G. Hanaoka, Y. Hanaoka, and H. Imai, "Parallel key-insulated public key encryption," in *Proceedings of the 9th International Workshop on Theory and Practice in Public Key Cryptography*, vol. 3958 of *Lecture Notes in Comput. Sci.*, pp. 105–122, Springer, Berlin, Germany, 2006.

[26] M. Bellare and A. Palacio, "Protecting against key-exposure: strongly key-insulated encryption with optimal threshold," *Applicable Algebra in Engineering, Communication and Computing*, vol. 16, no. 6, pp. 379–396, 2006.

[27] R. P. Li, J. Yu, J. Wang, G. W. Li, and D. X. Li, "Key-insulated group signature scheme with verifier-local revocation," in *Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pp. 273–278, IEEE, 2007.

[28] J. K. Liu and D. S. Wong, "Solutions to key exposure problem in ring signature," *International Journal of Network Security*, vol. 6, no. 2, pp. 170–180, 2008.

[29] Z.-M. Wan, X.-J. Lai, J. Weng, S.-L. Liu, Y. Long, and X. Hong, "Certificateless key-insulated signature without random oracles," *Journal of Zhejiang University: Science A*, vol. 10, no. 12, pp. 1790–1800, 2009.

[30] L. Liu and Z. Cao, "Analysis of two signature schemes from CIS'2006," in *Proceedings of the 2009 International Conference on Computational Intelligence and Security, CIS 2009*, pp. 405–408, December 2009.

[31] Z. Wan, X. Lai, J. Weng, S. Liu, and X. Hong, "Identity-based key-insulated proxy signature," *Journal of Electronics*, vol. 26, no. 6, pp. 853–858, 2009.

[32] H. Du, J. Li, Y. Zhang, T. Li, and Y. Zhang, "Certificate-Based Key-Insulated Signature," in *Data and Knowledge Engineering*, vol. 7696 of *Lecture Notes in Computer Science*, pp. 206–220, Springer, Berlin, Germany, 2012.

[33] J. Chen, K. Chen, Y. Wang, X. Li, Y. Long, and Z. Wan, "Identity-based key-insulated signcryption," *Informatica*, vol. 23, no. 1, pp. 27–45, 2012.

[34] J. Fan, Y. Zheng, and X. Tang, "Key-insulated signcryption," *Journal of Universal Computer Science*, vol. 19, no. 10, pp. 1351–1374, 2013.

[35] H. Wang, H. Cao, and D. Li, "Identity-based key-insulated signcryption scheme," *Journal of Computational Information Systems*, vol. 9, no. 8, pp. 3067–3075, 2013.

[36] H. Zhao, J. Yu, S. Duan, X. Cheng, and R. Hao, "Key-insulated aggregate signature," *Frontiers in Computer Science*, vol. 8, no. 5, pp. 837–846, 2014.

[37] J. Chen, Y. Long, K. Chen, and J. Guo, "Attribute-based key-insulated signature and its applications," *Information Sciences. An International Journal*, vol. 275, pp. 57–67, 2014.

[38] G. Zhu, H. Xiong, and Z. Qin, "Fully Secure Identity Based Key-Insulated Signcryption in the Standard Model," *Wireless Personal Communications*, vol. 79, no. 2, pp. 1401–1416, 2014.

[39] J. Li, H. Du, Y. Zhang, T. Li, and Y. Zhang, "Provably secure certificate-based key-insulated signature scheme," *Concurrency Computation Practice and Experience*, vol. 26, no. 8, pp. 1546–1560, 2014.

[40] H. Xiong, S. Wu, J. Geng, E. Ahene, S. Wu, and Z. Qin, "A pairing-free key-insulated certificate-based signature scheme with provable security," *KSII Transactions on Internet and Information Systems*, vol. 9, no. 3, pp. 1246–1259, 2015.

[41] Y. Lu, Q. Zhang, and J. Li, "An improved certificateless strong key-insulated signature scheme in the standard model," *Advances in Mathematics of Communications*, vol. 9, no. 3, pp. 353–373, 2015.

[42] J. Li, H. Du, and Y. Zhang, "Certificate-based key-insulated signature in the standard model," *Computer Journal*, vol. 59, no. 7, pp. 1028–1039, 2016.

[43] H. Hong and Z. Sun, "A Key-Insulated Ciphertext Policy Attribute Based Signcryption for Mobile Networks," *Wireless Personal Communications*, pp. 1–14, 2016.

[44] 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. 1000–1029, 2016.

[45] A. Boldyreva, A. Palacio, and B. Warinschi, "Secure proxy signature schemes for delegation of signing rights," *Journal of Cryptology. The Journal of the International Association for Cryptologic Research*, vol. 25, no. 1, pp. 57–115, 2012.

[46] Shamus Software Ltd. Miracl library, http://github.com/miracl/MIRACL.

[47] The Certicom Corporation. Sec 2: recommended elliptic curve domain parameters. The Standard for Efficient Cryptography Group, 2000.