INFORMATICA, 2022, Vol. 33, No. 1, 151–179
© 2022 *Vilnius University*
DOI: https://doi.org/10.15388/22-INFOR474

Leakage-Resilient Revocable Certificateless Encryption with an Outsourced Revocation Authority

Yuh-Min TSENG[1,*], Sen-Shan HUANG[1], Tung-Tso TSAI[2], Yun-Hsin CHUANG[1], Ying-Hao HUNG[1]

[1] *Department of Mathematics, National Changhua University of Education, Changhua 500, Taiwan*
[2] *Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung 202, Taiwan*
e-mail: ymtseng@cc.ncue.edu.tw

Received: March 2021; accepted: January 2022

Abstract. To resolve both certificate management and key escrow problems, a certificateless public-key system (CLPKS) has been proposed. However, a CLPKS setting must provide a revocation mechanism to revoke compromised users. Thus, a revocable certificateless public-key system (RCLPKS) was presented to address the revocation issue and, in such a system, the key generation centre (KGC) is responsible to run this revocation functionality. Furthermore, a RCLPKS setting with an outsourced revocation authority (ORA), named RCLPKS-ORA setting, was proposed to employ the ORA to alleviate the KGC's computational burden. Very recently it was noticed that adversaries may adopt side-channel attacks to threaten these existing conventional public-key systems (including CLPKS, RCLPKS and RCLPKS-ORA). Fortunately, leakage-resilient cryptography offers a solution to resist such attacks. In this article, the *first* leakage-resilient revocable certificateless encryption scheme with an ORA, termed LR-RCLE-ORA scheme, is proposed. The proposed scheme is formally shown to be semantically secure against three types of adversaries in the RCLPKS and RCLPKS-ORA settings while resisting side-channel attacks. In the proposed scheme, adversaries are allowed to continually extract partial ingredients of secret keys participated in various computational algorithms of the proposed scheme while retaining its security.

Key words: leakage-resilience, certificateless encryption, revocation, key encapsulation.

1. Introduction

To eliminate the management of both public keys and their associated certificates in the traditional public-key systems (PKS), an identity (ID)-based public-key system (IDPKS) was proposed (Boneh and Franklin, 2001). In an IDPKS setting, a private key generator (PKG) is responsible to generate all participants' secret keys. Hence, the IDPKS setting has an inborn disadvantage, namely, the key escrow problem in the sense that the PKG

[*]Corresponding author.

can decrypt any ciphertexts of all participants and sign any messages on behalf of all participants. To resolve both certificate management and key escrow problems, Al-Riyami and Paterson (2003) proposed the certificateless public-key system (CLPKS). In which, there are two kinds of participants, namely, a key generation center (KGC) and users. The KGC is responsible to generate all users' identity secret keys. In the meantime, each user chooses a personal secret key and sets the associated public key. Therefore, each user has two secret keys, namely, identity secret key and personal secret key. Obviously, the CLPKS setting can solve both the key escrow and certificate management problems.

Under some situations, a user's ID or public key has to be revoked before expiration. Although the certificate revocation list (CRL) (Housley *et al.*, 2002) is a well-known revocation method, it is unsuitable for IDPKS and CLPKS settings because no certificate is required. Therefore, an IDPKS or CLPKS setting must provide a revocation mechanism to revoke compromised users. Tseng and Tsai (2012) has presented a revocable IDPKS setting. By this revocable concept of Tseng and Tsai, revocable CLPKS (RCLPKS) settings (Shen *et al.*, 2014; Tsai and Tseng, 2015; Hung *et al.*, 2016) were presented to address the revocation issue and the key generation center (KGC) is also responsible to run this revocation functionality. Furthermore, a RCLPKS setting with an outsourced revocation authority (ORA), named RCLPKS-ORA setting (Tsai *et al.*, 2015; Du *et al.*, 2018), was presented to employ the ORA to alleviate the KGC's computational burden.

Typically, all public-key systems mentioned above have a nature assumption that secret (or private) keys are completely hidden to adversaries. However, a new type of attacks, called "side-channel attacks", has threatened these existing public-key systems. An adversary may employ side-channel attacks, such as power analysis (Kocher *et al.*, 1999) and timing attack (Kocher, 1996; Brumley and Boneh, 2005) to continually obtain partial ingredients of a participant's secret (or private) keys so that the associated cryptographic schemes/protocols could be broken. Ways to withstand side-channel attacks on cryptographic schemes/protocols have received significant attention of researchers. Fortunately, leakage-resilient cryptography offers a solution to resist such attacks. Up to date, little research addresses leakage-resilient certificateless public-key systems. In this paper, our aim is to propose the *first* leakage-resilient revocable certificateless encryption (LR-RCLE) scheme with an outsourced revocation authority (ORA), termed LR-RCLE-ORA scheme.

1.1. *Related Work*

In leakage-resilient cryptography, a cryptographic scheme/protocol remains secure even though partial ingredients of a participant's secret (or private) keys in this scheme/protocol is leaked to adversaries. For leakage property, there are two leakage models, namely, bounded and continual. In the bounded leakage model (Katz and Vaikuntanathan, 2009; Alwen *et al.*, 2009), total leaked bit sizes of secret keys for a cryptographic scheme/protocol are limited during its life cycle. Obviously, this limitation is impractical. In contrast, in the continual leakage model, adversaries may continually get leaked information of secret keys for each invocation of cryptographic scheme/protocol. Indeed, the continual leakage model is more accredited (Galindo and Virek, 2013; Wu *et al.*, 2019, 2020; Tseng *et al.*, 2020; Hsieh *et al.*, 2020; Tsai *et al.*, 2021) and it consists of four properties as follows:

- *Only computation leakage*: An adversary can extract partial ingredients of secret keys involved in the current computation round.
- *Bounded leakage of single computational algorithm*: A cryptographic scheme/protocol typically includes several computation rounds. In each computation round, an adversary can extract partial ingredients of secret keys. Namely, an adversary can select a leakage function f for each computation round and obtain the leaked information $f(SK)$, where SK denotes the involved secret keys and $f(SK)$ is bounded to λ bits.
- *Independent leakage*: Any two leaked partial ingredients of secret keys in various computation rounds are mutually independent. For achieving this property, a secret key must be updated before (or after) running each computation round.
- *Overall unbounded leakage*: The total amount of leaked information is overall unbounded. Indeed, by the independent leakage property, the total leakage amount of secret keys is unlimited.

According to the usage of secret key or public key, there are two kinds of encryption schemes, namely, symmetric encryption and asymmetric encryption based on various public-key systems. For a symmetric encryption scheme, a pre-shared secret key between a sender and a receiver is used to encrypt and decrypt procedures. A symmetric encryption scheme is typically employed to encrypt a large size of message and have high-throughput efficiency. On the contrary, for an asymmetric encryption scheme, a sender uses a designated receiver's public key to encrypt message while the receiver uses her/his private key to decrypt it. Generally, an asymmetric encryption scheme is employed to encrypt a short size of message (e.g. a session key) or authenticate identity/message so that the throughput of encryption/decryption procedure is not their priority. Also, for considering leakage-resilient property, there are two kinds of leakage-resilient encryption schemes, namely, leakage-resilient symmetric encryption and leakage-resilient asymmetric encryption based on various public-key systems.

Here, let's introduce the evolution of leakage-resilient symmetric encryption schemes. The first generic construction of leakage-resilient symmetric encryption based on minimal assumptions has been proposed by Hazay *et al.* (2013). However, the efficiency of Hazay *et al.*'s scheme is not good so that Abdalla *et al.* (2013) improved their scheme to propose an efficient leakage-resilient symmetric encryption scheme using the AES block cipher without constructing a leakage resilient block cipher. Recently, for enhancing the efficiency, several leakage-resilient authenticated symmetric encryption schemes based on hardware AES coprocessors (Unterstein *et al.*, 2020; Bronchain *et al.*, 2021) have been proposed.

In the following, several leakage-resilient asymmetric encryption (or key encapsulation) schemes based on traditional PKS, IDPKS and CLPKS settings are reviewed. Based on a traditional PKS setting, the first leakage-resilient encryption (LRE) scheme was presented by Akavia *et al.* (2009). Subsequently, several LRE schemes (Naor and Segev, 2009, 2012; Liu *et al.*, 2013; Li *et al.*, 2013) were proposed to improve security and performance of Akavia *et al.*'s scheme. However, all these LRE schemes above are secure only in the bounded leakage model. Moreover, Kiltz and Pietrzak (2010) presented the first LRE scheme under the continual leakage model. Furthermore, Galindo *et al.* (2016) presented an efficient ElGamal-like LRE scheme under the continual leakage model. Based on an IDPKS setting, Brakerski *et al.* (2010) proposed the first leakage-resilient ID-based encryption (LR-IBE) scheme under the continual leakage model. Subsequently, several LR-IBE schemes (Yuen *et al.*, 2012; Li *et al.*, 2016) were also proposed to improve security and performance of Brakerski *et al.*'s scheme.

Up to date, little research addresses leakage-resilient certificateless public-key systems. In 2013, the first leakage-resilient certificateless encryption (LR-CLE) scheme was presented by Xiong *et al.* (2013). To improve the security and performance of Xiong *et al.*'s scheme, Zhou *et al.* (2016) proposed a new leakage-resilient certificateless signcryption scheme. However, both Zhou *et al.*'s and Xiong *et al.*'s schemes are secure only under the bounded leakage model. In 2018, Wu *et al.* (2018) proposed the *first* LR-CLE scheme under the continual leakage model. In the generic bilinear group (GBG) model (Boneh *et al.*), Wu *et al.*'s LR-CLE scheme is semantically secure against chosen ciphertext attacks for two adversaries, namely, outsider and honest-but-curious KGC.

1.2. *Contribution and Organization*

As mentioned earlier, a RCLPKS setting with an outsourced revocation authority (ORA), named RCLPKS-ORA setting, can revoke compromised users and alleviate the KGC's revocation computation burden. However, up to date, there are no leakage-resilient revocable certificateless encryption (LR-RCLE) or leakage-resilient revocable certificateless key encapsulation scheme. In this article, the *first* leakage-resilient revocable certificateless encryption scheme with an ORA, termed LR-RCLE-ORA scheme, is proposed. By extending the adversary models of the revocable certificateless encryption (RCLE) (Tsai and Tseng, 2015) and leakage resilience certificateless encryption (LR-CLE) schemes (Xiong *et al.*, 2013; Wu *et al.*, 2018), a new adversary model of LR-RCLE-ORA schemes is presented. Under this new adversary model, the proposed scheme is formally shown to be semantically secure against three types of adversaries, namely, outsider, revoked user and honest-but-curious KGC. Finally, comparisons with previously proposed schemes (Tsai and Tseng, 2015; Xiong *et al.*, 2013; Wu *et al.*, 2018), the proposed scheme has the following merits. (1) It can resist side-channel attacks and has leakage resilience properties. (2) The revocation functionality is outsourced to an ORA to alleviate the computational load of the KGC. (3) It permits adversaries to continually extract partial ingredient of secret keys and offers the overall unbounded leakage property.

The remaining paper is organized as below. Several preliminaries are presented in Section 2. In Section 3, the syntax (framework) and adversary model (security notions) of LR-RCLE-ORA schemes are defined. The proposed LR-RCLE-ORA scheme is presented in Section 4. In Section 5, the security of the proposed scheme is formally established. The comparisons of the proposed scheme with some RCLE and LR-CLE schemes are given in Section 6. Conclusions are drawn in Section 7.

2. Preliminaries

2.1. *Bilinear Groups*

Let G_1 be an additive group of a prime order p and Q be a generator of G_1. Let G_2 be a multiplicative group of the same order p. A bilinear admissible pairing $\hat{e} : G_1 \times G_1 \rightarrow G_2$ possesses the following properties:

- Bilinear: for $r, s \in Z_p^*$, $\hat{e}(r \cdot Q, s \cdot Q) = \hat{e}(Q, Q)^{rs}$;
- Non-degenerate: $\hat{e}(Q, Q) \neq 1$;
- Efficiently computable: for $R, S \in G_1$, $\hat{e}(R, S)$ is efficiently computed.

We say that $\{G_1, G_2, p, Q, \hat{e}\}$ are the bilinear group parameters. A reader may refer to Boneh and Franklin (2001), Scott (2011) for detailed definitions of bilinear groups.

2.2. *Generic Bilinear Group Model*

In 2005, Boneh *et al.* (2005) defined the generic bilinear group (GBG) model, which is a technique of proving the security of some cryptographic schemes. In the GBG model, the discrete logarithm problems on groups of a large order would be solved if collisions of the groups were found by adversaries after finishing the security games of the cryptographic scheme.

In the GBG model, as mentioned in Section 2.1, let $\{G_1, G_2, p, Q, \hat{e}\}$ be the bilinear group parameters and let each element in two groups G_1 and G_2 be represented by distinct bit-strings. In such a case, a random injective mapping $\Phi_1 : Z_p^* \rightarrow \xi G_1$ is used to encode all elements of G_1, where ξG_1 denotes the set of the encoded bit-strings of G_1. By the same reason, the other random injective mapping $\Phi_2 : Z_p^* \rightarrow \xi G_2$ is employed to encode all elements of G_2, where ξG_2 denotes the set of the encoded bit-strings of G_2. Two sets ξG_1 and ξG_2 satisfy $|\xi G_1| = |\xi G_2| = p$ and $\xi G_1 \cap \xi G_2 = \phi$. In addition, let three operations O_1, O_2 and O_p, respectively, denote the addition of G_1, the multiplication of G_2 and the bilinear pairing $\hat{e}$. To perform these group operations, an adversary must request the associated operations (oracles) O_1, O_2 and O_p which are defined as below:

- $O_1(\Phi_1(r), \Phi_1(s)) \rightarrow \Phi_1(r + s \bmod p)$;
- $O_2(\Phi_2(r), \Phi_2(s)) \rightarrow \Phi_2(r + s \bmod p)$;
- $O_p(\Phi_1(r), \Phi_1(s)) \rightarrow \Phi_2(rs \bmod p)$.

Note that $r, s \in Z_p^*$, $Q = \Phi_1(1)$ and $\hat{e}(Q, Q) = \Phi_2(1)$.

After finishing the security games in the GBG model of a cryptographic scheme, the discrete logarithm problem on G_1 or G_2 would be resolved if adversaries discovered a collision on G_1 or G_2. The discrete logarithm problem and assumption are presented as below:

- **Discrete logarithm (DL) problem and assumption:** Let $\{G_1, G_2, p, Q, \hat{e}\}$ be the bilinear group parameters. Given a group element $r \cdot Q \in G_1$ or $\hat{e}(Q, Q)^r \in G_2$ for unknown $r \in Z_p^*$, the DL problem is to find r from $r \cdot Q$ or $\hat{e}(Q, Q)^r$. The DL assumption is that no polynomial time algorithm A with non-negligible probability can discover r.

2.3. *The Security Measure of Leaked Information*

To measure the security influence incurred by leaked information of secret keys (finite random variables) involved in a cryptographic scheme, we first introduce the concept of entropy. The entropy of a random variable is employed to denote its uncertainty for guessing this random variable. Let R be a finite random variable and let $\Pr[R = r]$ denote the associated probability of $R = r$. In the following, we present two kinds of min-entropies:

1. Min-entropy of R:

$$H_\infty(R) = -\log_2\Big(\max_r \Pr[R = r]\Big);$$

2. Average conditional min-entropy of R under the condition $S = s$:

$$\widetilde{H}_\infty(R|S = s) = -\log_2\Big(E_{s\leftarrow S}\Big[\max_r \Pr[R = r|S = s]\Big]\Big).$$

In 2008, Dodis *et al.* (2008) inferred the following consequence related to the security influence incurred by leaked information of a secret key (finite random variable).

Lemma 1. *Let λ denote the maximal bit-length of leaked information of a secret key (a finite random variable) R. Let $f : R \rightarrow \{0,1\}^\lambda$ denote the associated leakage function of R. Under the condition $f(R)$, the average conditional min-entropy on R is $\widetilde{H}_\infty(R|f(R)) \geqq H_\infty(R) - \lambda$.*

Galindo and Virek (2013) further presented the following consequences related to multiple secret keys (finite random variables).

Lemma 2. *Let $R_1, R_2, \dots, R_n$ be n random variables. Let $F \in Z_p[R_1, R_2, \dots, R_n]$ be a polynomial with at most degree d. Let P_k denote a probability distribution on Z_p such that $H_\infty(P_k) \geqq \log p - \lambda$ and $0 \leqq \lambda \leqq \log p$, for $k = 1, 2, \dots, n$. If all $R_k = r_k \leftarrow Z_p$ with probability distribution P_k are mutually independent, we have* $\Pr[F(R_1 = r_1, R_2 = r_2, \dots, R_n = r_n) = 0] \leqq (d/p)2^\lambda$.

Corollary 1. $\Pr[F(R_1 = r_1, R_2 = r_2, \dots, R_n = r_n) = 0]$ *is negligible if $\lambda < (1 - \epsilon)\log p$, where ϵ is a positive value.*

3. Syntax and Adversary Model

The syntax (framework) and adversary model (security notions) of LR-RCLE-ORA schemes are presented as follows.

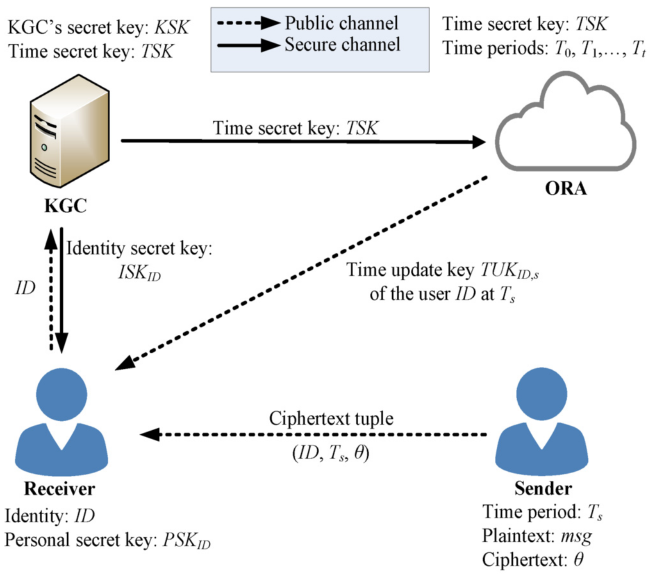

Fig. 1. The system architecture of a LR-RCLE-ORA scheme.

3.1. *Syntax of LR-RCLE-ORA schemes*

Here, let us present the system architecture of an LR-RCLE-ORA scheme as depicted in Fig. 1. An LR-RCLE-ORA scheme consists of three roles, namely, a key generation centre (KGC), an outsourced revocation authority (ORA) and users (senders and receivers). Each user with an identity ID randomly selects a personal secret key PSK_{ID} by herself/himself. For each user with ID, the KGC with a secret key KSK is responsible to generate and securely send an identity secret key ISK_{ID} to the user. For each non-revoked user with ID at each period T_s, the ORA with a time secret key TSK is responsible to generate and publicly send a time update key $TUK_{ID,s}$ to the user. For compromised or revoked users, the ORA will not generate any associated time update keys. At period T_s, when a sender would like to encrypt a plaintext msg to a receiver with ID, the sender uses the receiver's ID and associated public keys to encrypt msg to generate a ciphertext tuple (ID, T_s, θ) while sending (ID, T_s, θ) to the receiver. The receiver then uses her/his PSK_{ID}, ISK_{ID} and $TUK_{ID,s}$ to decrypt (ID, T_s, θ) to obtain msg.

In the following, we summarize some notations used in the whole paper:

- KSK: the KGC's secret key;
- KPK: the KGC's public key;
- TSK: the time secret key;
- TPK: the time public key;
- ID: a user's identity;
- PSK_{ID}: a user ID's personal secret key;
- PPK_{ID}: a user ID's personal public key;

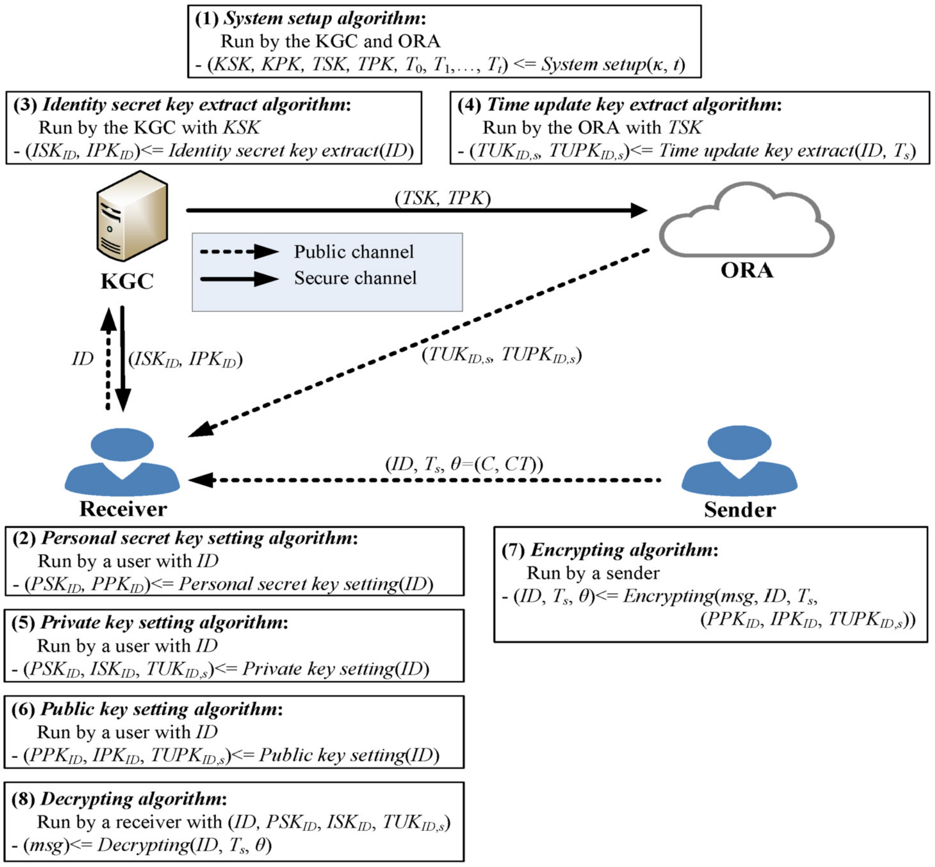
(1) System setup algorithm:
Run by the KGC and ORA
- $(KSK, KPK, TSK, TPK, T_0, T_1, \dots, T_t)$ <= System setup(κ, t)
(3) Identity secret key extract algorithm:
Run by the KGC with KSK
- (ISK_{ID}, IPK_{ID})<= Identity secret key extract(ID)
(4) Time update key extract algorithm:
Run by the ORA with TSK
- $(TUK_{ID,s}, TUPK_{ID,s})$<= Time update key extract(ID, T_s)
(TSK, TPK)
Public channel
Secure channel
KGC
ORA
ID
(ISK_{ID}, IPK_{ID})
$(TUK_{ID,s}, TUPK_{ID,s})$
$(ID, T_s, \theta=(C, CT))$
Receiver
Sender
(2) Personal secret key setting algorithm:
Run by a user with ID
- (PSK_{ID}, PPK_{ID})<= Personal secret key setting(ID)
(5) Private key setting algorithm:
Run by a user with ID
- $(PSK_{ID}, ISK_{ID}, TUK_{ID,s})$<= Private key setting(ID)
(6) Public key setting algorithm:
Run by a user with ID
- $(PPK_{ID}, IPK_{ID}, TUPK_{ID,s})$<= Public key setting(ID)
(8) Decrypting algorithm:
Run by a receiver with $(ID, PSK_{ID}, ISK_{ID}, TUK_{ID,s})$
- (msg)<= Decrypting(ID, T_s, θ)
(7) Encrypting algorithm:
Run by a sender
- (ID, T_s, θ)<= Encrypting(msg, ID, T_s, $(PPK_{ID}, IPK_{ID}, TUPK_{ID,s})$)

Fig. 2. The algorithm architecture of the proposed LR-RCLE-ORA scheme.

- ISK_{ID}: a user ID's identity secret key;
- IPK_{ID}: a user ID's identity public key;
- T_s: a period $T_s \in \{0, 1\}^*$, for $s = 1, \dots, t$, where t denotes the period length;
- $TUK_{ID,s}$: a user ID's time update key at period T_s;
- $TUPK_{ID,s}$: a user ID's time update public key at period T_s;
- msg: a message;
- θ: a ciphertext.

Based on the syntax (framework) in (Tsai and Tseng, 2015; Xiong *et al.*, 2013; Wu *et al.*, 2018), the syntax of LR-RCLE-ORA schemes is defined as follows. An LR-RCLE-ORA scheme consists of three roles, namely, a key generation centre (KGC), an outsourced revocation authority (ORA) and users (senders and receivers) while including eight algorithms: (1) *System setup*; (2) *Personal secret key setting*; (3) *Identity secret key extract*; (4) *Time update key extract*; (5) *Private key setting*; (6) *Public key setting*; (7) *Encrypting*; (8) *Decrypting*. Fig. 2 depicts the algorithm architecture, interactions and their inputs/outputs of the proposed LR-RCLE-ORA scheme. Eight algorithms of the LR-RCLE-ORA scheme are presented in Definition 1 as below:

Definition 1. An LR-RCLE-ORA scheme consists of three roles, namely, a key generation centre (KGC), an outsourced revocation authority (ORA) and users (senders and receivers). Eight algorithms of the LR-RCLE-ORA scheme are presented as below:

– *System setup*: By taking as input a security parameter κ and a period number t, the KGC generates the KGC's secret key $KSK = (KSK_{0,1}, KSK_{0,2})$ and associated public key KPK, the time secret key $TSK = (TSK_{0,1}, TSK_{0,2})$, and time public key TPK. The KGC then securely sends TSK to the ORA. Finally, the KGC publishes t periods $T_1, T_2, \dots, T_t$ and public system parameters PSP.
– *Personal secret key setting*: Each user with an identity ID randomly selects a personal secret key $PSK_{ID} = (PSK_{ID,0,1}, PSK_{ID,0,2})$ and generates the associated personal public key PPK_{ID}.
– *Identity secret key extract*: In this algorithm's i-th round, by taking as input a user ID and $(KSK_{i-1,1}, KSK_{i-1,2})$, the KGC first carries out two sub-algorithms ISKExtract-1 $(ID, KSK_{i-1,1})$ and ISKExtract-2 $(KSK_{i-1,2})$ to set the new KGC's secret key $(KSK_{i,1}, KSK_{i,2})$, and generate the user's identity secret key ISK_{ID} and associated identity public key IPK_{ID}. Finally, the KGC securely sends IPK_{ID} and ISK_{ID} to the user.
– *Time update key extract*: In this algorithm's j-th round, by taking as input a user ID, a period T_s and $(TSK_{j-1,1}, TSK_{j-1,2})$, the ORA carries out two sub-algorithms TUKExtract-1 $(ID, T_s, TSK_{j-1,1})$ and TUKExtract-2 $(TSK_{j-1,2})$ to set the new time secret key $(TSK_{j,1}, TSK_{j,2})$, and generate the user's time update key $TUK_{ID,s}$ and associated time update public key $TUPK_{ID,s}$ at period T_s. Finally, the ORA sends $TUK_{ID,s}$ and $TUPK_{ID,s}$ to the user.
– *Private key setting*: At period T_s, a user ID's private key tuple includes three parts, namely, PSK_{ID}, ISK_{ID}, and $TUK_{ID,s}$. The user also sets $PSK_{ID} = (PSK_{ID,0,1}, PSK_{ID,0,2})$ and $ISK_{ID} = (ISK_{ID,0,1}, ISK_{ID,0,2})$.
– *Public key setting*: At period T_s, a user ID's public key tuple includes three parts, namely, PPK_{ID}, IPK_{ID}, and $TUPK_{ID,s}$.
– *Encrypting*: At period T_s, by taking as input a plaintext msg and a receiver ID with public key tuple $(PPK_{ID}, IPK_{ID}, TUPK_{ID,s})$, the sender generates a ciphertext tuple $(ID, T_s, \theta = (C, CT = E_{EK}(msg)))$, where $E_{EK}(\cdot)$ is a symmetric encryption function and EK is an encryption key encrypted to generate C. Finally, the sender returns the ciphertext tuple $(ID, T_s, \theta = (C, CT))$.
– *Decrypting*: In this algorithm's k-th round, by taking as input $(ID, T_s, \theta = (C, CT))$, a receiver with ID uses her/his private key tuple $(PSK_{ID} = (PSK_{ID,k-1,1}, PSK_{ID,k-1,2}), ISK_{ID} = (ISK_{ID,k-1,1}, ISK_{ID,k-1,2}), TUK_{ID,s})$ to carry out two sub-algorithms DEC-1$(PSK_{ID,k-1,1}, ISK_{ID,k-1,1})$ and DEC-2$(PSK_{ID,k-1,2}, ISK_{ID,k-1,2}, TUK_{ID,s}, T_s, \theta = (C, CT))$ to set new private key tuple $(PSK_{ID} = (PSK_{ID,k,1}, PSK_{ID,k,2}), ISK_{ID} = (ISK_{ID,k,1}, ISK_{ID,k,2}), TUK_{ID,s})$, compute the encryption key EK from C, and decrypt $msg = D_{EK}(CT)$, where $D_{EK}(\cdot)$ is the corresponding symmetric decryption function of $E_{EK}(\cdot)$.

3.2. *Adversary Model of LR-RCLE-ORA Schemes*

By the entropy concept of secret keys mentioned in Section 2.3, six leakage functions $f_{ISKE,i}$, $h_{ISKE,i}$, $f_{TUKE,j}$, $h_{TUKE,j}$, $f_{DEC,k}$ and $h_{DEC,k}$ are employed to present capabilities of adversaries obtaining leaked information of secrets keys. Both $f_{ISKE,i}$ and $h_{ISKE,i}$ are employed to obtain leaked information of the KGC's secret key ($KSK_{i,1}$, $KSK_{i,2}$) used in the *Identity secret key extract* algorithm's i-th round. Both $f_{TUKE,j}$ and $h_{TUKE,j}$ are employed to obtain leaked information of the time secret key ($TSK_{j,1}$, $TSK_{j,2}$) used in the *Time update key extract* algorithm's j-th round. Furthermore, both $f_{DEC,k}$ and $h_{DEC,k}$ are employed to obtain leaked information of a receiver's private key tuple ($PSK_{ID} = (PSK_{ID,k,1}, PSK_{ID,k,2}), ISK_{ID} = (ISK_{ID,k,1}, ISK_{ID,k,2}), TUK_{ID,s}$) used in the *Decrypting* algorithm's k-th round of a user *ID*. An adversary can obtain at most λ bits of secret keys used in each associated algorithm, where λ is related to the security parameter selected in the System setup algorithm. Namely, $|f_{ISKE,i}|$, $|h_{ISKE,i}|$, $|f_{TUKE,j}|$, $|h_{TUKE,j}|$, $|f_{DEC,k}|$, $|h_{DEC,k}| \leqq \lambda$, where $|\cdot|$ denotes the output bit-length of a function. The leaked information of a leakage function f is denoted by Λf. In the sequel, leaked information of the six leakage functions are denoted as below:

- $\Lambda f_{ISKE,i} = f_{ISKE,i}(KSK_{i,1})$;
- $\Lambda h_{ISKE,i} = h_{ISKE,i}(KSK_{i,2})$;
- $\Lambda f_{TUKE,j} = f_{TUKE,j}(TSK_{j,1})$;
- $\Lambda h_{TUKE,j} = h_{TUKE,j}(TSK_{j,2})$;
- $\Lambda f_{DEC,k} = f_{DEC,k}(PSK_{ID,k,1}, ISK_{ID,k,1})$;
- $\Lambda h_{DEC,k} = h_{DEC,k}(PSK_{ID,k,2}, ISK_{ID,k,2}, TUK_{ID,k,2})$.

By extending the adversary model (security notions) in Tsai and Tseng (2015), Xiong *et al.* (2013), Wu *et al.* (2018), the adversary model of LR-RCLE-ORA schemes consists of three types of adversaries, namely, outsider (Type I, A_I), revoked user (Type II, A_{II}) and honest-but-curious KGC (Type III, A_{III}).

- Outsider (A_I): Although A_I is not a legal user of the LR-RCLE-ORA scheme, it may obtain the time update key $TUK_{ID,s}$ of any user *ID* at any period T_s from public channels. Also, A_I may get the personal secret key PSK_{ID} and the identity secret key ISK_{ID} of any user *ID*, but it is disallowed to get the identity secret key ISK_{ID^*} of the target user ID^*. For leakage resilient property, A_I can obtain leaked information of both the target user's $ISK_{ID^*} = (ISK_{ID^*,k,1}, ISK_{ID^*,k,2})$ used in the *Decrypting* algorithm's k-th round of the target user and the KGC's secret key ($KSK_{i,1}$, $KSK_{i,2}$) used in the *Identity secret key extract* algorithm's i-th round.
- Revoked user (A_{II}): When A_{II} is a legal user of the LR-RCLE-ORA scheme who has been revoked, it knows the personal secret key PSK_{ID} and the identity secret key ISK_{ID} of any user *ID*. Also, A_{II} may get the time update key $TUK_{ID,s}$ of any user *ID* at any period T_s, except TUK_{ID^*,s^*} of the target user ID^* at target period T_{s^*}. For leakage resilient property, A_{II} can obtain leaked information of the time secret key ($TSK_{j,1}$, $TSK_{j,2}$) used in the *Time update key extract* algorithm's j-th round.
- Honest-but-curious KGC (A_{III}): Since A_{III} knows the KGC's secret key *KSK* and time secret key *TSK*, it can get the identity secret key ISK_{ID} and time update key $TUK_{ID,s}$ of

any user ID at any period T_s. Also, A_{III} may get the personal secret key PSK_{ID} of any user ID, except PSK_{ID^*} of the target user ID^*. For leakage resilient property, A_{III} can obtain leaked information of the target user's $PSK_{ID^*} = (PSK_{ID^*,k,1}, PSK_{ID^*,k,2})$ used in the *Decrypting* algorithm's k-th round of the target user.

In the continual model of leakage resilient property, the adversary model of LR-RCLE-ORA schemes is defined by the following security game $G_{LR\text{-}RCLE\text{-}ORA}$ played by both an adversary (A_I, A_{II} or A_{III}) and a challenger B.

Definition 2. In the continual leakage model, an LR-RCLE-ORA scheme is semantically secure against chosen cipher-text attacks if no adversary (A_I, A_{II} or A_{III}) with non-negligible advantage wins the security game $G_{LR\text{-}RCLE\text{-}ORA}$ in polynomial time. This security game $G_{LR\text{-}RCLE\text{-}ORA}$ consists of three phases:

– *Setup phase*: By taking as input a security parameter κ and a period number t, a challenger B carries out the System setup algorithm of the LR-RCLE-ORA scheme to generate the KGC's secret key $KSK = (KSK_{0,1}, KSK_{0,2})$, the KGC's public key KPK, the time secret key $TSK = (TSK_{0,1}, TSK_{0,2})$, and the time public key TPK. Also, B sets t periods $T_1, T_2, \ldots, T_t$ and publishes public system parameters PSP. Additionally, B sends messages to the adversary of various types by the following rules:
  - If the adversary is of A_I, B sends out TSK;
  - If the adversary is of A_{II}, B sends out KSK;
  - If the adversary is of A_{III}, B sends out KSK and TSK.
– *Query phase*: In the phase, the adversary may request the following queries to B adaptively.
  - *Identity secret key query* (ID): For this query's i-th round, B sets the new KGC's secret key $(KSK_{i,1}, KSK_{i,2})$ by using $(KSK_{i-1,1}, KSK_{i-1,2})$. Also, B uses $(KSK_{i,1}, KSK_{i,2})$ to generate and return the associated identity secret key ISK_{ID} and identity public key IPK_{ID}.
  - *Identity secret key leak query* $(f_{ISKE,i}, h_{ISKE,i}, i)$: In the Identity secret key query's i-th round, this query is allowed to be requested only once. B returns leaked information $(\Lambda f_{ISKE,i}, \Lambda h_{ISKE,i})$.
  - *Time update key query* (ID, T_s): In this query's j-th round, B sets the new time secret key $(TSK_{j,1}, TSK_{j,2})$ by using $(TSK_{j-1,1}, TSK_{j-1,2})$. Also, B uses $(TSK_{j,1}, TSK_{j,2})$, ID and T_s to generate and return the associated time update key $TUK_{ID,s}$ and the time update public key $TUPK_{ID,s}$.
  - *Time update key leak query* $(f_{TUKE,j}, h_{TUKE,j}, j)$: In the Time update key query's j-th round, this query is allowed to be requested only once. B returns leaked information $(\Lambda f_{TUKE,j}, \Lambda h_{TUKE,j})$.
  - *Public key retrieve query* (ID, T_s): B returns the associated public key tuple $(PPK_{ID}, IPK_{ID}, TUPK_{ID,s})$.
  - *Public key replace query* (ID, T_s, $(PPK'_{ID}, IPK'_{ID}, TUPK'_{ID,s})$): B sets the new public key tuple $(PPK'_{ID}, IPK'_{ID}, TUPK'_{ID,s})$ of the user ID at period T_s.
  - *Personal secret key corrupt query* (ID): If the Public key replace query (ID) has never been requested, B returns the associated personal secret key PSK_{ID}.

- *Decrypting query* $(ID, T_s, \theta = (C, CT))$: In this query's k-th round, B sets the user ID's new private key tuple $(PSK_{ID} = (PSK_{ID,k,1}, PSK_{ID,k,2}), ISK_{ID} = (ISK_{ID,k,1}, ISK_{ID,k,2}), TUK_{ID,s})$ by using $(PSK_{ID} = (PSK_{ID,k-1,1}, PSK_{ID,k-1,2}), ISKID = (ISK_{ID,k-1,1}, ISK_{ID,k-1,2}), TUK_{ID,s})$. Also, B uses the new private key tuple to compute the encryption key EK from C, and decrypt $msg = D_{EK}(CT)$. B returns msg.
- *Decrypting leak query* $(ID, T_s, f_{DEC,k}, h_{DEC,k}, k)$: In the Decrypting query's k-th round of the user ID at period T_s, this query is allowed to be requested only once. B returns leaked information $(\Lambda f_{DEC,k}, \Lambda h_{DEC,k})$.

– *Challenge phase*. The adversary sends a target identity ID^*, a target period T_{s^*} and a plaintext pair (msg_0^*, msg_1^*) to B. B chooses an unbiased random bit $b \in \{0, 1\}$ and carries out the Encrypting algorithm with $(ID^*, T_{s^*}, (PPK_{ID^*}, IPK_{ID^*}, TUPK_{ID^*,s^*}), msg_b^*)$ to generate C^*, EK^* and $CT^* = E_{EK^*}(msg_b^*)$. Finally, B returns the ciphertext tuple $(ID^*, T_{s^*}, \theta = (C^*, CT^*))$ to the adversary. Additionally, the associated conditions must be satisfied according to various types of adversaries:

1. If the adversary is of A_I, the Identity secret key query (ID^*) has never been requested;
2. If the adversary is of A_{II}, the Time update key query (ID^*, T_s^*) has never been requested;
3. If the adversary is of A_{III}, both the Personal secret key corrupt query (ID^*) and the Public key replace query $(ID^*, T_s^*, (PPK'_{ID^*}, IPK'_{ID^*}, TUPK'_{ID^*,s^*}))$ has never been requested.

– *Guess phase*. In this phase, the adversary must output a bit $b' \in \{0, 1\}$. If $b' = b$, we say that it wins the game $G_{LR\text{-}RCLE\text{-}ORA}$ and its advantage is $|\Pr[b' = b] - 1/2|$.

4. The Proposed LR-RCLE-ORA Scheme

The proposed LR-RCLE-ORA scheme consists of eight algorithms as follows:

– *System setup*: By taking as input a security parameter κ and a period number t, the KGC chooses the bilinear group parameters $\{G_1, G_2, p, Q, \hat{e}\}$, picks a symmetric encryption function $E(\cdot)$ and its associated decryption function $D(\cdot)$, and sets a period set $T = \{T_0, T_1, T_2, \ldots, T_t\}$. The KGC carries out the following procedures:

(1) Randomly select an integer $\alpha \in Z_p^*$, and generate the KGC's secret key $KSK = \alpha \cdot Q$ and public key $KPK = \hat{e}(Q, KSK)$. Additionally, the KGC randomly selects an integer $x_0 \in Z_p^*$ and generates the KGC's current secret key $(KSK_{0,1}, KSK_{0,2}) = (x_0 \cdot Q, KSK + (-x_0) \cdot Q)$.

(2) Randomly select an integer $\beta \in Z_p^*$, and generate the time secret key $TSK = \beta \cdot Q$ and time public key $TPK = \hat{e}(Q, TSK)$. Additionally, the KGC securely sends TSK to the ORA. The ORA then selects a random integer $y_0 \in Z_p^*$ and generates the current time secret key $(TSK_{0,1}, TSK_{0,2}) = (y_0 \cdot Q, TSK + (-y_0) \cdot Q)$.

(3) Randomly select four integers $m, n, r, s \in Z_p^*$, and compute $M = m \cdot Q$, $N = n \cdot Q$, $R = r \cdot Q$ and $S = s \cdot Q$.
(4) Publish public system parameters $PSP = \{G_1, G_2, p, Q, \hat{e}, KPK, TPK, T, D(), E(), M, N, R, S\}$.

– *Personal secret key setting*: Each user with an identity *ID* randomly selects an integer $\gamma \in Z_p^*$ and generates personal secret key $PSK_{ID} = \gamma \cdot Q$ and the associated personal public key $PPK_{ID} = \hat{e}(Q, PSK_{ID})$.
– *Identity secret key extract*: In this algorithm's i-th round, by taking as input a user's *ID* and $(KSK_{i-1,1}, KSK_{i-1,2})$, the KGC carries out two sub-algorithms ISKExtract-1$(ID, KSK_{i-1,1})$ and ISKExtract-2$(KSK_{i-1,2})$ as below:
  - ISKExtract-1$(ID, KSK_{i-1,1})$:
    (1) Randomly select an integer $x_i \in Z_p^*$, and compute $KSK_{i,1} = KSK_{i-1,1} + x_i \cdot Q$.
    (2) Randomly select an integer $u \in Z_p^*$, and compute $IPK_{ID} = u \cdot Q$ and temporary value $TV_{ISKE} = KSK_{i,1} + u \cdot (M + ID \cdot N)$.
  - ISKExtract-2$(KSK_{i-1,2})$:
    (1) Compute $KSK_{i,2} = KSK_{i-1,2} + (-x_i) \cdot Q$ and $ISK_{ID} = KSK_{i,2} + TV_{ISKE}$.
    (2) Send the user's identity secret key ISK_{ID} and associated identity public key IPK_{ID} to the user using a secure channel.

– *Time update key extract*: In this algorithm's j-th round, by taking as input a user's *ID*, a period T_s and $(TSK_{j-1,1}, TSK_{j-1,2})$, the ORA carries out two sub-algorithms TUKExtract-1$(ID, T_s, TSK_{j-1,1})$ and TUKExtract-2$(TSK_{j-1,2})$ as below:
  - TUKExtract-1$(ID, T_s, TSK_{j-1,1})$:
    (1) Randomly select an integer $y_j \in Z_p^*$, and compute $TSK_{j,1} = TSK_{j-1,1} + y_j \cdot Q$.
    (2) Randomly select an integer $v \in Z_p^*$, and compute $TUPK_{ID,s} = v \cdot Q$ and temporary value $TV_{TUKE} = TSK_{j,1} + v \cdot (R + (ID||T_s) \cdot S)$.
  - TUKExtract-2$(TSK_{j-1,2})$:
    (1) Compute $TSK_{j,2} = TSK_{j-1,2} + (-y_j) \cdot Q$ and $TUK_{ID,s} = TSK_{j,2} + TV_{TUKE}$.
    (2) Send the user's time update key $TUK_{ID,s}$ and associated time update public key $TUPK_{ID,s}$ to the user.

– *Private key setting*: At period T_s, a user *ID*'s private key tuple includes three parts, namely, PSK_{ID}, ISK_{ID}, and $TUK_{ID,s}$. The user carries out the following procedures:
  (1) Randomly select an integer $z_0 \in Z_p^*$ and compute the current personal secret key $(PSK_{ID,0,1}, PSK_{ID,0,2}) = (z_0 \cdot Q, PSK_{ID} + (-z_0) \cdot Q)$.
  (2) Randomly select an integer $w_0 \in Z_p^*$ and compute the current identity secret key $(ISK_{ID,0,1}, ISK_{ID,0,2}) = (w_0 \cdot Q, ISK_{ID} + (-w_0) \cdot Q)$.
  (3) Set the user's private key tuple $(PSK_{ID} = (SK_{ID,0,1}, SK_{ID,0,2}), ISK_{ID} = (ISK_{ID,0,1}, ISK_{ID,0,2}), TUK_{ID,s})$.
– *Public key setting*: At period T_s, a user *ID* sets her/his public key tuple $(PPK_{ID}, IPK_{ID}, TUPK_{ID,s})$.
– *Encrypting*: At period T_s, by taking as input a plaintext *msg* and a receiver *ID* with public key tuple $(PPK_{ID}, IPK_{ID}, TUPK_{ID,s})$, the sender carries out the following procedures:

(1) Randomly select an integer $ek \in Z_p^*$.
(2) Compute $C = ek \cdot Q$, $K_1 = (PPK_{ID})^{ek}$, $K_2 = (KPK \cdot \hat{e}(IPK_{ID}, M + ID \cdot N))^{ek}$ and $K_3 = (TPK \cdot \hat{e}(TUPK_{ID,s}, R + (ID||T_s) \cdot S))^{ek}$.
(3) Set the encryption key $EK = K_1 \oplus K_2 \oplus K_3$.
(4) Compute $CT = E_{EK}(msg)$ and return the ciphertext tuple $(ID, T_s, \theta = (C, CT))$.

– *Decrypting*: In this algorithm's k-th round, by taking as input $(ID, T_s, \theta = (C, CT))$, a receiver ID uses her/his private key tuple $(PSK_{ID} = (PSK_{ID,k-1,1}, PSK_{ID,k-1,2})$, $ISK_{ID} = (ISK_{ID,k-1,1}, ISK_{ID,k-1,2}), TUK_{ID,s})$ to carry out two sub-algorithms DEC-1$(PSK_{ID,k-1,1}, ISK_{ID,k-1,1})$ and DEC-2$(PSK_{ID,k-1,2}, ISK_{ID,k-1,2}, TUK_{ID,s}, T_s, \theta = (C, CT))$ as below:

• DEC-1$(PSK_{ID,k-1,1}, ISK_{ID,k-1,1})$
(1) Randomly select an integer $z_k \in Z_p^*$, and compute $PSK_{ID,k,1} = PSK_{ID,k-1,1} + z_k \cdot Q$.
(2) Randomly select an integer $w_k \in Z_p^*$, and compute $ISK_{ID,k,1} = ISK_{ID,k-1,1} + w_k \cdot Q$.
(3) Compute two temporary values $TV_1 = \hat{e}(C, PSK_{ID,k,1})$ and $TV_2 = \hat{e}(C, ISK_{ID,k,1})$.

• DEC-2$(PSK_{ID,k-1,2}, ISK_{ID,k-1,2}, TUK_{ID,s}, T_s, \theta = (C, CT))$
(1) Compute $PSK_{ID,k,2} = PSK_{ID,k-1,2} + (-z_k) \cdot Q$ and $ISK_{ID,k,2} = ISK_{ID,k-1,2} + (-w_k) \cdot Q$.
(2) Compute $K_1' = TV_1 \cdot \hat{e}(C, PSK_{ID,k,2})$, $K_2' = TV_2 \cdot \hat{e}(C, ISK_{ID,k,2})$ and $K_3' = \hat{e}(C, TUK_{ID,s})$.
(3) Compute the encryption key $EK' = K_1' \oplus K_2' \oplus K_3'$.
(4) Decrypt the plaintext $msg = D_{EK'}(CT)$.

In the following, by the key refreshing technique, we have

$$\begin{aligned}
KSK &= KSK_{0,1} + KSK_{0,2} = KSK_{1,1} + KSK_{1,2} = \cdots = KSK_{i,1} + KSK_{i,2};\\
TSK &= TSK_{0,1} + TSK_{0,2} = TSK_{1,1} + TSK_{1,2} = \cdots = TSK_{j,1} + TSK_{j,2};\\
PSK_{ID} &= PSK_{ID,0,1} + PSK_{ID,0,2} = PSK_{ID,1,1} + PSK_{ID,1,2}\\
&\quad = \cdots = PSK_{ID,k,1} + PSK_{ID,k,2};\\
ISK_{ID} &= ISK_{ID,0,1} + ISK_{ID,0,2} = ISK_{ID,1,1} + ISK_{ID,1,2}\\
&\quad = \cdots = ISK_{ID,k,1} + ISK_{ID,k,2}.
\end{aligned}$$

In the following, we show the correctness of $EK = K_1 \oplus K_2 \oplus K_3 = K_1' \oplus K_2' \oplus K_3' = EK'$.

$$\begin{aligned}
EK &= K_1 \oplus K_2 \oplus K_3\\
&= (PPK_{ID})^{ek} \oplus \big(KPK \cdot \hat{e}(IPK_{ID}, M + ID \cdot N)\big)^{ek}\\
&\quad \oplus \big(TPK \cdot \hat{e}\big(TUPK_{ID,s}, R + (ID||T_s) \cdot S\big)\big)^{ek}\\
&= \hat{e}(Q, PSK_{ID})^{ek} \oplus \big(\hat{e}(Q, KSK) \cdot \hat{e}(u \cdot Q, M + ID \cdot N)\big)^{ek}\\
&\quad \oplus \big(\hat{e}(Q, TSK) \cdot \hat{e}\big(v \cdot Q, R + (ID||T_s) \cdot S\big)\big)^{ek}\\
&= \hat{e}(Q, PSK_{ID})^{ek} \oplus \big(\hat{e}(Q, KSK) \cdot \hat{e}\big(Q, u \cdot (M + ID \cdot N)\big)\big)^{ek}
\end{aligned}$$

$$
\begin{aligned}
&\oplus \big(\hat{e}(Q, TSK) \cdot \hat{e}\big(Q, v \cdot \big(R + (ID||T_s) \cdot S\big)\big)\big)^{ek} \\
&= \hat{e}(Q, PSK_{ID})^{ek} \oplus \big(\hat{e}\big(Q, KSK + u \cdot (M + ID \cdot N)\big)\big)^{ek} \\
&\quad \oplus \big(\hat{e}\big(Q, TSK + v \cdot \big(R + (ID||T_s) \cdot S\big)\big)\big)^{ek} \\
&= \hat{e}(ek \cdot Q, PSK_{ID}) \oplus \hat{e}\big(ek \cdot Q, KSK + u \cdot (M + ID \cdot N)\big) \\
&\quad \oplus \hat{e}\big(ek \cdot Q, TSK + v \cdot \big(R + (ID||T_s) \cdot S\big)\big) \\
&= \hat{e}(C, PSK_{ID}) \oplus \hat{e}\big(C, KSK + u \cdot (M + ID \cdot N)\big) \\
&\quad \oplus \hat{e}\big(C, TSK + v \cdot \big(R + (ID||T_s) \cdot S\big)\big) \\
&= \hat{e}(C, PSK_{ID}) \oplus \hat{e}\big(C, KSK_{i,1} + KSK_{i,2} + u \cdot (M + ID \cdot N)\big) \\
&\quad \oplus \hat{e}\big(C, TSK_{j,1} + TSK_{j,2} + v \cdot \big(R + (ID||T_s) \cdot S\big)\big) \\
&= \hat{e}(C, PSK_{ID}) \oplus \hat{e}(C, KSK_{i,2} + TV_{ISKE}) \oplus \hat{e}(C, TSK_{j,2} + TV_{TUKE}) \\
&= \hat{e}(C, PSK_{ID}) \oplus \hat{e}(C, ISK_{ID}) \oplus \hat{e}(C, TUK_{ID,s}) \\
&= \hat{e}(C, PSK_{ID,k,1} + PSK_{ID,k,2}) \oplus \hat{e}(C, ISK_{ID,k,1} + ISK_{ID,k,2}) \oplus \hat{e}(C, TUK_{ID,s}) \\
&= \big(\hat{e}(C, PSK_{ID,k,1}) \cdot \hat{e}(C, PSK_{ID,k,2})\big) \\
&\quad \oplus \big(\hat{e}(C, ISK_{ID,k,1}) \cdot \hat{e}(C, ISK_{ID,k,2})\big) \oplus \hat{e}(C, TUK_{ID,s}) \\
&= \big(TV_1 \cdot \hat{e}(C, PSK_{ID,k,2})\big) \oplus \big(TV_2 \cdot \hat{e}(C, ISK_{ID,k,2})\big) \oplus \hat{e}(C, TUK_{ID,s}) \\
&= K_1' \oplus K_2' \oplus K_3' = EK'.
\end{aligned}
$$

5. Security Analysis

As the security game $G_{LR\text{-}RCLE\text{-}ORA}$ presented in Definition 2, the adversary model consists of three types of adversaries, namely, outsider (Type I, A_I), revoked user (Type II, A_{II}) and honest-but-curious KGC (Type III, A_{III}). Under the GBG model presented in Section 2, we employ three theorems to demonstrate that the proposed LR-RCLE-ORA scheme is semantically secure against chosen cipher-text attacks against three types of adversaries, respectively. The relationship and robustness of security analysis are depicted in Fig. 3. In Theorem 1, we first discuss the advantage (denoted by Adv_{A-I-W}) of an outsider (A_I) without requesting any leak queries and then evaluate the advantage (denoted by Adv_{A-I}) of an outsider (A_I) with requesting *Identity secret key leak* and *Decrypting leak queries*. Indeed, by Corollary 1, Adv_{A-I} is negligible based on the DL assumption. For Theorems 2 and 3, by similar arguments as in Theorem 1, the advantages (denoted by Adv_{A-II} and Adv_{A-III}) of a revoked user (A_{II}) and an honest-but-curious KGC (A_{III}) are also negligible based on the DL assumption.

Theorem 1. *In the GBG model, the proposed LR-RCLE-ORA scheme is semantically secure against chosen cipher-text attacks against an outsider* (A_I) *in the security game* $G_{LR\text{-}RCLE\text{-}ORA}$.

Proof. Let A_I be an outsider of the security game $G_{LR\text{-}RCLE\text{-}ORA}$ played with a challenger B. In the GBG model, A_I may request three queries (oracles) O_1, O_2 and O_p to

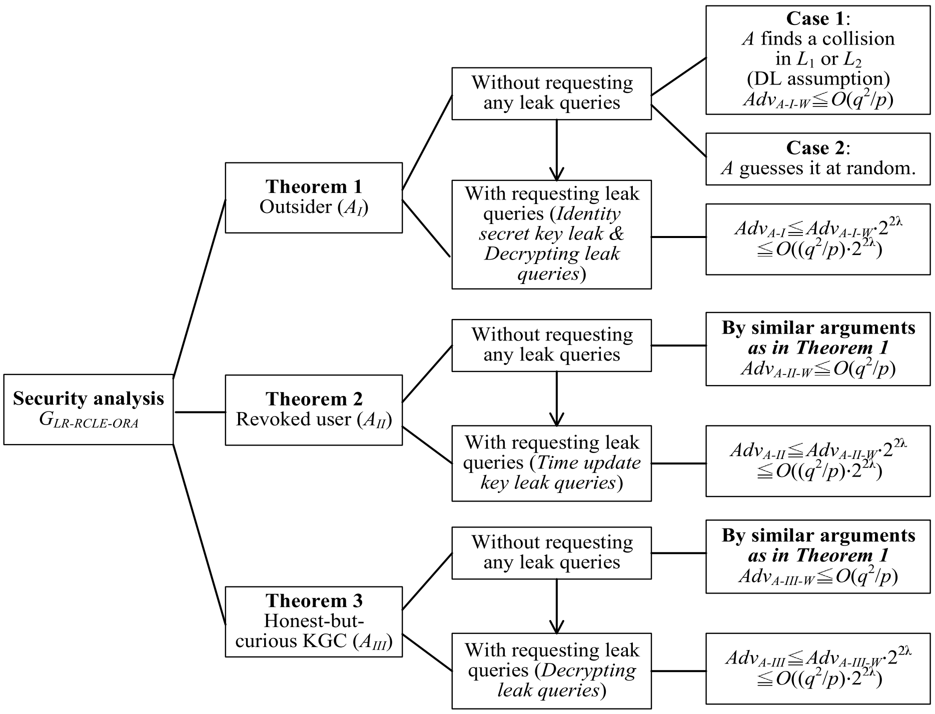
Security analysis
$G_{LR\text{-}RCLE\text{-}ORA}$
Theorem 1
Outsider (A_I)
Theorem 2
Revoked user (A_{II})
Theorem 3
Honest-but-curious KGC (A_{III})
Without requesting any leak queries
With requesting leak queries (Identity secret key leak & Decrypting leak queries)
With requesting leak queries (Time update key leak queries)
With requesting leak queries (Decrypting leak queries)
Case 1: A finds a collision in L_1 or L_2 (DL assumption) $Adv_{A\text{-}I\text{-}W} \leqq O(q^2/p)$
Case 2: A guesses it at random.
$Adv_{A\text{-}I} \leqq Adv_{A\text{-}I\text{-}W} \cdot 2^{2\lambda} \leqq O((q^2/p) \cdot 2^{2\lambda})$
By similar arguments as in Theorem 1 $Adv_{A\text{-}II\text{-}W} \leqq O(q^2/p)$
$Adv_{A\text{-}II} \leqq Adv_{A\text{-}II\text{-}W} \cdot 2^{2\lambda} \leqq O((q^2/p) \cdot 2^{2\lambda})$
By similar arguments as in Theorem 1 $Adv_{A\text{-}III\text{-}W} \leqq O(q^2/p)$
$Adv_{A\text{-}III} \leqq Adv_{A\text{-}III\text{-}W} \cdot 2^{2\lambda} \leqq O((q^2/p) \cdot 2^{2\lambda})$

Fig. 3. The relationship and robustness of security analysis for the proposed scheme.

perform three corresponding group operations. In the security game $G_{LR\text{-}RCLE\text{-}ORA}$, four phases are presented as below:

– *Setup phase*: By taking as input a security parameter κ and a period number t, B carries out the *System setup* algorithm of the LR-RCLE-ORA scheme to generate the KGC's secret key $KSK = (KSK_{0,1}, KSK_{0,2})$, the KGC's public key KPK, the time secret key TSK, and the time pubic key TPK. Also, B sets a period set $T = \{T_0, T_1, T_2, \ldots, T_t\}$ and publishes public system parameters $PSP = \{G_1, G_2, p, Q, \hat{e}, KPK, TPK, T, D(), E(), M, N, R, S\}$. Additionally, B sends TSK to A_I since A_I is an outsider. To respond the queries requested by A_I, five initially empty lists L_1, L_2, L_{ISK}, L_{TUK} and L_{PSK} are constructed as below:
  - L_1 and L_2 are used to encode elements of groups G_1 and G_2, respectively.
    (1) L_1 includes pairs of $(\varXi G_{1,t,u,v}, \varTheta G_{1,t,u,v})$. An element in G_1 is represented by a multivariate polynomial $\varXi G_{1,t,u,v}$ and $\varTheta G_{1,t,u,v}$ is the corresponding encoded bit-string, where t, u and v, respectively, denote the query type t, the u-th query and the v-th element in G_1. B initially adds seven pairs $(\varXi Q, \varTheta G_{1,I,0,1})$, $(\varXi KSK, \varTheta G_{1,I,0,2})$, $(\varXi TSK, \varTheta G_{1,I,0,3})$, $(\varXi M, \varTheta G_{1,I,0,4})$, $(\varXi N, \varTheta G_{1,I,0,5})$, $(\varXi R, \varTheta G_{1,I,0,6})$ and $(\varXi S, \varTheta G_{1,I,0,7})$ in L_1.
    (2) L_2 includes pairs of $(\varXi G_{2,t,u,v}, \varTheta G_{2,t,u,v})$. An element in G_2 is represented by a multivariate polynomial $\varXi G_{2,t,u,v}$ and $\varTheta G_{2,t,u,v}$ is the corresponding encoded

bit-string, where t, u and v have the same meanings with those indices in L_1. B initially adds two pairs $(\Xi KPK, \Theta G_{2,I,0,1})$ and $(\Xi TPK, \Theta G_{2,I,0,2})$ in L_2, where $\Xi KPK = \Xi Q \cdot \Xi KSK$ and $\Xi TPK = \Xi Q \cdot \Xi TSK$.

It is worth mentioning the two transforming rules defined below.

(1) By taking as input a polynomial $\Xi G_{1,t,u,v}/\Xi G_{2,t,u,v}$, B looks for $(\Xi G_{1,t,u,v}, \Theta G_{1,t,u,v})/(\Xi G_{2,t,u,v}, \Theta G_{2,t,u,v})$ in L_1/L_2. If it is found, B returns the encoded bit-string $\Theta G_{1,t,u,v}/\Theta G_{2,t,u,v}$. Otherwise, B randomly chooses and returns a distinct encoded bit-string $\Theta G_{1,t,u,v}/\Theta G_{2,t,u,v}$. In addition, B adds $(\Xi G_{1,t,u,v}, \Theta G_{1,t,u,v})/(\Xi G_{2,t,u,v}, \Theta G_{2,t,u,v})$ in L_1/L_2.

(2) By taking as input an encoded bit-string $\Theta G_{1,t,u,v}/\Theta G_{2,t,u,v}$, B looks for $(\Xi G_{1,t,u,v}, \Theta G_{1,t,u,v})/(\Xi G_{2,t,u,v}, \Theta G_{2,t,u,v})$ in L_1/L_2. If it is found, B returns the associated multivariate polynomial $\Xi G_{1,t,u,v}/\Xi G_{2,t,u,v}$. Otherwise, B terminates the game.

- L_{ISK} includes tuples of $(ID, \Xi ISK_{ID}, \Xi IPK_{ID})$, where ΞISK_{ID} and ΞIPK_{ID}, respectively, represent the user's ISK_{ID} and IPK_{ID}.
- L_{PSK} includes tuples of $(ID, \Xi PSK_{ID}, \Xi PPK_{ID})$, where ΞPSK_{ID} and ΞPPK_{ID}, respectively, denote the user's PSK_{ID} and PPK_{ID}.
- L_{TUK} includes tuples of $(ID, T_s, \Xi TUK_{ID,s}, \Xi TUPK_{ID,s})$, where $\Xi TUK_{ID,s}$ and $\Xi TUPK_{ID,s}$, respectively, denote the user's $TUK_{ID,s}$ and $TUPK_{ID,s}$.

Finally, these public system parameters ΞQ, ΞM, ΞN, ΞR, ΞS, ΞKPK and ΞTPK are sent to A_I. Also, B sends the time secret key ΞTSK to A_I.

– *Query phase*: A_I can adaptively request various queries to B at most q times. Note that A_I does not need to request the *Time update key leak query* and *Public key replace query*, since A_I may get the time update key $TUK_{ID,s}$ of any user ID at any period T_s from public channels.

- O_1 *query* $(\Theta G_{1,Q,l,1}, \Theta G_{1,Q,l,2}, OP)$: In this query's l-th round, B carries out the following steps:
  (1) Get a pair of polynomials $(\Xi G_{1,Q,l,1}, \Xi G_{1,Q,l,2})$ by transforming a pair of bit-strings $(\Theta G_{1,Q,l,1}, \Theta G_{1,Q,l,2})$ in L_1.
  (2) Compute the polynomial $\Xi G_{1,Q,l,3} = \Xi G_{1,Q,l,1} + \Xi G_{1,Q,l,2}$ if $OP =$ "addition", and $\Xi G_{1,Q,l,3} = \Xi G_{1,Q,l,1} - \Xi G_{1,Q,l,2}$ if $OP =$ "subtraction".
  (3) Return the encoded bit-string $\Theta G_{1,Q,l,3}$ by transforming $\Xi G_{1,Q,l,3}$ in L_1.
- O_2 *query* $(\Theta G_{2,Q,l,1}, \Theta G_{2,Q,l,2}, OP)$: In this query's l-th round, B carries out the following steps:
  (1) Get a pair of polynomials $(\Xi G_{2,Q,l,1}, \Xi G_{2,Q,l,2})$ by transforming a pair of bit-strings $(\Theta G_{2,Q,l,1}, \Theta G_{2,Q,l,2})$ in L_2.
  (2) Compute the polynomial $\Xi G_{2,Q,l,3} = \Xi G_{2,Q,l,1} + \Xi G_{2,Q,l,2}$ if $OP =$ "multiplication", and $\Xi G_{2,Q,l,3} = \Xi G_{2,Q,l,1} - \Xi G_{2,Q,l,2}$ if $OP =$ "division".
  (3) Return the encoded bit-string $\Theta G_{2,Q,l,3}$ by transforming $\Xi G_{2,Q,l,3}$ in L_2.
- O_p *query* $(\Theta G_{1,P,l,1}, \Theta G_{1,P,l,2})$: In this query's l-th round, B carries out the following steps:
  (1) Get a pair of polynomials $(\Xi G_{1,P,l,1}, \Xi G_{1,P,l,2})$ by transforming a pair of bit-strings $(\Theta G_{1,P,l,1}, \Theta G_{1,P,l,2})$ in L_1.

(2) Compute the polynomial $\Xi G_{2,P,l,1} = \Xi G_{1,P,l,1} \cdot \Xi G_{1,P,l,2}$.
(3) Return the encoded bit-string $\Theta G_{2,P,l,1}$ by transforming $\Xi G_{2,P,l,1}$ in L_2.

- *Identity secret key query* (ID): In this query's i-th round, B searches (ID, ΞISK_{ID}, ΞIPK_{ID}) in L_{ISK}. If it is found, B transforms (ΞISK_{ID}, ΞIPK_{ID}) to send two encoded bit-strings (ΘISK_{ID}, ΘIPK_{ID}) to A_I. Otherwise, B carries out the following steps:
  (1) Choose a new variate $\Xi TG_{ISK,i,1}$ in G_1.
  (2) Set two polynomials $\Xi IPK_{ID} = \Xi TG_{ISK,i,1}$ and $\Xi TID = ID$.
  (3) Set the user's identity secret key $\Xi ISK_{ID} = \Xi KSK + \Xi TG_{ISK,i,1} \cdot (\Xi M + \Xi N \cdot \Xi TID)$ while adding (ID, ΞISK_{ID}, ΞIPK_{ID}) in L_{ISK}.
  (4) Transform (ΞISK_{ID}, ΞIPK_{ID}) to send two encoded bit-strings (ΘISK_{ID}, ΘIPK_{ID}) to A_I.
- *Identity secret key leak query* ($f_{ISKE,i}$, $h_{ISKE,i}$, i): In the *Identity secret key query*'s i-th round, this query is allowed to be requested only once. B returns leaked information ($\Lambda f_{ISKE,i}$, $\Lambda h_{ISKE,i}$), where $\Lambda f_{ISKE,i} = f_{ISKE,i}(KSK_{i,1})$ and $\Lambda h_{ISKE,i} = h_{ISKE,i}(KSK_{i,2})$.
- *Time update key query* (ID, T_s): In this query's j-th round, B searches (ID, T_s, $\Xi TUK_{ID,s}$, $\Xi TUPK_{ID,s}$) in L_{TUK}. If it is found, B transforms ($\Xi TUK_{ID,s}$, $\Xi TUPK_{ID,s}$) to return two encoded bit-strings ($\Theta TUK_{ID,s}$, $\Theta TUPK_{ID,s}$). Otherwise, B carries out the following steps:
  (1) Choose a new variate $\Xi TG_{TUK,ID,j,1}$ in G_1.
  (2) Set a polynomial $\Xi TUPK_{ID,s} = \Xi TG_{TUK,ID,j,1}$ and $\Xi TTD = ID||T_s$.
  (3) Set the user's time update key $\Xi TUK_{ID,t} = \Xi TSK + \Xi TG_{TUK,ID,j,1} \cdot (\Xi R + \Xi S \cdot \Xi TTD)$ while adding (ID, T_s, $\Xi TUK_{ID,s}$, $\Xi TUPK_{ID,s}$) in L_{TUK}.
  (4) Transform ($\Xi TUK_{ID,s}$, $\Xi TUPK_{ID,s}$) to return two encoded bit-strings ($\Theta TUK_{ID,s}$, $\Theta TUPK_{ID,s}$).
- *Time update key leak query* ($f_{TUKE,j}$, $h_{TUKE,j}$, j): In the *Time update key query*'s j-th round, this query is allowed to be requested only once. B returns leaked information ($\Lambda f_{TUKE,j}$, $\Lambda h_{TUKE,j}$), where $\Lambda f_{TUKE,j} = f_{TUKE,j}(TSK_{j,1})$ and $\Lambda h_{TUKE,j} = h_{TUKE,j}(TSK_{j,2})$.
- *Public key retrieve query* (ID, T_s): B applies ID and T_s to search L_{ISK}, L_{PSK} and L_{TUK} to get the associated public key tuple (ΞPPK_{ID}, ΞIPK_{ID}, $\Xi TUPK_{ID,s}$). B returns the corresponding tuple (ΘPPK_{ID}, ΘIPK_{ID}, $\Theta TUPK_{ID,s}$).
- *Public key replace query* (ID, T_s, ($\Theta PPK'_{ID}$, $\Theta IPK'_{ID}$, $\Theta TUPK'_{ID,s}$)): B first transforms a tuple of bit-strings ($\Theta PPK'_{ID}$, $\Theta IPK'_{ID}$, $\Theta TUPK'_{ID,s}$) to get the corresponding tuple of polynomials ($\Xi PPK'_{ID}$, $\Xi IPK'_{ID}$, $\Xi TUPK'_{ID,s}$). B replaces the related tuples with (ID, $-$, $\Xi PPK'_{ID}$) in L_{PSK}, (ID, $-$, IPK'_{ID}) in L_{ISK}, and (ID, T_s, $-$, $\Xi TUPK'_{ID,s}$) in L_{TUK}.
- *Personal secret key corrupt query* (ID): If the *Public key replace query* (ID) has never been requested, B uses ID to search (ID, ΞPSK_{ID}, ΞPPK_{ID}) in L_{PSK}. If it is found, B transforms (ΞPSK_{ID}, ΞPPK_{ID}) to return two encoded bit-strings (ΘPSK_{ID}, ΘPPK_{ID}). Otherwise, B carries out the following steps:
  (1) Choose a new variate $\Xi TG_{PSK,ID,1}$ in G_1.
  (2) Set two polynomials $\Xi PSK_{ID} = \Xi TG_{PSK,ID,1}$ and $\Xi PPK_{ID} = \Xi Q \cdot \Xi PSK_{ID}$, and add ($ID$, ΞPSK_{ID}, ΞPPK_{ID}) in L_{PSK}.

(3) Transform (ΞPSK_{ID}, ΞPPK_{ID}) to return two encoded bit-strings (ΘPSK_{ID}, ΘPPK_{ID}).

- *Decrypting query* ($ID, T_s, \theta = (C, CT)$): In this query's k-th round, B carries out the following steps:
  (1) By ID and T_s, B finds the associated private key tuple ($\Xi PSK_{ID}, \Xi ISK_{ID}, \Xi TUK_{ID,s}$) in L_{PSK}, L_{ISK} and L_{TUK}.
  (2) B transforms the ciphertext C to the polynomial ΞC in L_1 and sets three polynomials $\Xi K_1 = \Xi PSK_{ID} \cdot \Xi C$, $\Xi K_2 = \Xi ISK_{ID} \cdot \Xi C$ and $\Xi K_3 = \Xi TUK_{ID,s} \cdot \Xi C$. Moreover, B transforms ΞK_1, ΞK_2 and ΞK_3 to obtain bit-strings ΘK_1, ΘK_2 and ΘK_3, respectively. Hence, B can gain the encryption key $\Theta EK = \Theta K_1 \oplus \Theta K_2 \oplus \Theta K_3$. Finally, B returns the encryption key ΘEK and the plaintext $msg = D_{\Theta EK}(CT)$.
- *Decrypting leak query* ($ID, T_s, f_{DEC,k}, h_{DEC,k}, k$): In the *Decrypting query*'s k-th round of the user ID at period T_s, this query is allowed to be requested only once. B returns leaked information ($\Lambda f_{DEC,k}$, $\Lambda h_{DEC,k}$), where $\Lambda f_{DEC,k} = f_{DEC,k}(PSK_{ID,k,1}, ISK_{ID,k,1})$ and $\Lambda h_{DEC,k} = h_{DEC,k}(PSK_{ID,k,2}, ISK_{ID,k,2}, TUK_{ID,s})$.

– *Challenge phase.* A_I sends a target identity ID^*, a target period T_{s^*} and a plaintext pair (msg_0^*, msg_1^*) to B. Here the *Identity secret key query* (ID^*) must have never been requested by A_I. B chooses an unbiased random bit $b \in \{0, 1\}$ and carries out the *Encrypting* algorithm with ($ID^*, T_{s^*}, (PPK_{ID^*}, IPK_{ID^*}, TUPK_{ID^*,s^*}), msg_b^*$) to generate C^*, EK^* and $CT^* = E_{EK^*}(msg_b^*)$. Finally, B returns the ciphertext tuple ($ID^*, T_{s^*}, \theta = (C^*, CT^*)$) to the adversary.

– *Guess phase*. In this phase, A_I must output a bit $b' \in \{0, 1\}$. If $b' = b$, we say that it wins the game $G_{LR\text{-}RCLE\text{-}ORA}$ and its advantage is $|\Pr[b' = b] - 1/2|$.

To evaluate the advantage that A_I wins the game $G_{LR\text{-}RCLE\text{-}ORA}$, we count the total number of elements in both L_1 and L_2. Subsequently, we count the maximal degrees of polynomials in L_1 and L_2, respectively.

■ **The total number of elements in both L_1 and L_2:**

- 7 and 2 elements are increased in L_1 and L_2, respectively, in the *Setup phase*.
- For each O_1, O_2 or O_p *query*, at most 3 elements are increased in L_1 or L_2.
- For each *Identity secret key query*, at most 3 elements are increased in L_1.
- For each *Time update key query*, at most 3 elements are increased in L_1.
- For each *Decrypting query*, at most 4 elements are increased in L_1.

Let q_O denote the total number of O_1, O_2 and O_p queries. Let q_I, q_T and q_D, respectively, be the query numbers of the *Identity secret key query*, *Time update key query* and *Decrypting query*. Since A_I is allowed to request all queries at most q times, we have

$$|L_1| + |L_2| \leqq 9 + 3q_O + 3q_I + 3q_T + 4q_D \leqq 4q.$$

■ **The maximal degrees of polynomials in L_1 and L_2:**

(1) The maximal degree of polynomials in L_1 is 3 due to the following facts:

- In the *Setup phase*, 7 new variates (polynomials) $\varXi Q$, $\varXi KSK$, $\varXi TSK$, $\varXi M$, $\varXi N$, $\varXi R$ and $\varXi S$ are initially increased in L_1. All these polynomials have degree 1.
- For the O_1 *query*, $\varXi G_{1,Q,l,3}$ has the maximal degree of $\varXi G_{1,Q,l,1}$ and $\varXi G_{1,Q,l,2}$.
- For the *Identity secret key query*, $\varXi TG_{ISK,i,1}$, $\varXi TID$ and $\varXi ISK_{ID}$ have degrees 1, 1 and 3, respectively.
- For the *Time update key query*, $\varXi TG_{TUK,ID,j,1}$, $\varXi TTD$ and $\varXi TUK_{ID,t}$ have degrees 1, 1 and 3, respectively.

(2) The maximal degree of polynomials in L_2 is 6 by the following facts:

- In the *Setup* phase, $\varXi KPK$ and $\varXi TPK$ have degree 2.
- For the O_2 *query*, $\varXi G_{2,Q,l,3}$ has the maximal degree of $\varXi G_{2,Q,l,1}$ and $\varXi G_{2,Q,l,2}$.
- For the O_p *query*, $\varXi G_{2,P,l,1}$ has degree at most 6 because $\varXi G_{2,P,l,1} = \varXi G_{1,P,l,1} \cdot \varXi G_{1,P,l,2}$, and both $\varXi G_{1,P,l,1}$ and $\varXi G_{1,P,l,2}$ belong to L_1.
- For the *Decrypting query*, all $\varXi K_1$, $\varXi K_2$ and $\varXi K_3$ have degrees 2.

In the following, let us first evaluate the advantage that A_I wins $G_{LR\text{-}RCLE\text{-}ORA}$ without requesting any leak query. Subsequently, the advantage of A_I is evaluated when it is allowed to request two kinds of leak queries (*Identity secret key leak query* and *Decrypting leak query*).

■ **The advantage of A_I without requesting any leak query:** If either of the following two cases occurs, A_I wins the game.

Case 1: A_I discovers a collision of two elements in L_1 or L_2. Let us first evaluate the collision probability in L_1. Let n be the number of all variates in L_1 and B selects n random values $v_l \in Z_p^*$ for $l = 1, 2, \dots, n$. Let $\varXi G_{1,i}$ and $\varXi G_{1,j}$ be any two distinct polynomials in L_1. B then computes $\varXi G_{1,C}(v_1, v_2, \dots, v_n) = \varXi G_{1,i} - \varXi G_{1,j}$. If $\varXi G_{1,C}(v_1, v_2, \dots, v_n) = 0$, it is said that the collision occurs. By Lemma 2, the probability of $\varXi G_{1,C}(v_1, v_2, \dots, v_n) = 0$ is at most $3/p$ because the maximal polynomial degree in L_1 is 3 and no information ($\lambda = 0$) is leaked. Since there are $\binom{|L_1|}{2}$ distinct pairs $(\varXi G_{1,i}, \varXi G_{1,j})$ in L_1, the collision probability is $(3/p)\binom{|L_1|}{2}$. Similarly, the collision probability in L_2 is $(6/p)\binom{|L_2|}{2}$. As mentioned earlier, we have $|L_1| + |L_2| \leqq 4q$. Let the probability of Case 1 be denoted by Pr[Case 1]. Then we have

$$\Pr[\text{Case } 1] \leqq (3/p)\binom{|L_1|}{2} + (6/p)\binom{|L_2|}{2} \leqq (6/p)\big(|L_1| + |L_2|\big)^2 \leqq 96q^2/p.$$

Case 2: If Case 1 does not occur and A_I gets no leaked information, the success probability of $b' = b$ is $1/2$ in the *Guess* phase. Let Pr[Case 2] denote the success probability that Case 2 occurs. Then we have Pr[Case 2] $\leqq 1/2$.

Let $\Pr_{A-I-W}$ and Adv_{A-I-W} be the probability and advantage, respectively, that A_I wins the game without requesting any leak query. By Pr[Case 1] and Pr[Case 2], we have

$$\Pr_{A-I-W} \leqq \Pr[\text{Case 1}] + \Pr[\text{Case 2}] \leqq 96q^2/p + (1/2),$$
$$Adv_{A-I-W} \leqq \left|96q^2/p + (1/2) - (1/2)\right| = 96q^2/p = O\left(q^2/p\right).$$

Hence, Adv_{A-I-W} is negligible if $q = poly(\log p)$.

- **The advantage of A_I with requesting two kinds of leak queries:** A_I can obtain leaked information of related secret keys by the *Identity secret key leak query* and *Decrypting leak query*.
  1. *Identity secret key leak query* $(f_{ISKE,i}, h_{ISKE,i}, i)$: By this query, A_I may derive leaked information $(\Lambda f_{ISKE,i}, \Lambda h_{ISKE,i})$ such that $|f_{ISKE,i}|$, $|h_{ISKE,i}| \leqq \lambda$, where $\Lambda f_{ISKE,i} = f_{ISKE,i}(KSK_{i,1})$ and $\Lambda h_{ISKE,i} = h_{ISKE,i}(KSK_{i,2})$ that are discussed as below:
     - $(KSK_{i,1}, KSK_{i,2})$: Indeed, the KGC's secret key KSK has the property in the sense that $KSK = KSK_{0,1} + KSK_{0,2} = KSK_{1,1} + KSK_{1,2} = \cdots = KSK_{i,1} + KSK_{i,2}$. By the refreshing technique, leaked information of $KSK_{i-1,1}/KSK_{i-1,2}$ is independent of that of $KSK_{i,1}/KSK_{i,2}$. Thus, A_I may derive at most 2λ bits of KSK.
  2. *Decrypting leak query* $(ID, T_s, f_{DEC,k}, h_{DEC,k}, k)$: As mentioned earlier, A_I is not a legal user of the LR-RCLE-ORA scheme, but it may get the time update key $TUK_{ID,s}$ of any user ID at any period T_s from public channels. Also, A_I may get the personal secret key PSK_{ID} and the identity secret key ISK_{ID} of any user ID, but it is disallowed to get the identity secret key ISK_{ID^*} of the target user ID^*. Therefore, by this query, A_I may derive leaked information $(\Lambda f_{DEC,k}, \Lambda h_{DEC,k})$, where $\Lambda f_{DEC,k} = f_{DEC,k}(ISK_{ID^*,k,1})$ and $\Lambda h_{DEC,k} = h_{DEC,k}(ISK_{ID^*,k,2})$ that are discussed as below:
     - $(ISK_{ID^*,k,1}, ISK_{ID^*,k-1,2})$: Indeed, the identity secret key ISK_{ID^*} satisfies $ISK_{ID^*} = ISK_{ID^*,0,1} + ISK_{ID^*,0,2} = ISK_{ID^*,1,1} + ISK_{ID^*,1,2} = \cdots = ISK_{ID^*,k,1} + ISK_{ID^*,k,2}$. By the refreshing technique, leaked information of $ISK_{ID^*,k-1,1}/ISK_{ID^*,k-1,2}$ is independent of that of $ISK_{ID^*,k,1}/ISK_{ID^*,k,2}$. Thus, A_I may derive at most 2λ bits of ISK_{ID^*}.

  Let $\Pr_{A-I}$ and Adv_{A-I} be the probability and advantage, respectively, that A_I wins $G_{LR\text{-}RCLE\text{-}ORA}$ with requesting the *Identity secret key leak query* and *Decrypting leak query*. If A_I can obtain the KGC's secret key KSK or the target user's identity key ISK_{ID^*}, A_I may decrypt the message. Three events are defined as below:
  1. Let $EKSK$ denote the event that A_I gets the KSK by $\Lambda f_{ISKE,i}$ and $\Lambda h_{ISKE,i}$. Additionally, $\overline{EKSK}$ is the complement event of $EKSK$.
  2. Let $EISK$ that A_I gets the ISK_{ID^*} by $\Lambda f_{DEC,k}$ and $\Lambda h_{DEC,k}$. Additionally, $\overline{EISK}$ is the complement event of $EISK$.
  3. Let ECB denote the event that A_I outputs a correct b'.

  Hence, the advantage $\Pr_{A-I}$ is $\Pr[ECB]$ and the following inequality holds:

$$\begin{aligned}\Pr_{A-I} &= \Pr[ECB]\\ &= \Pr\left[ECB \wedge (EKSK \vee EISK)\right] + \Pr\left[ECB \wedge (\overline{EKSK} \wedge \overline{EISK})\right]\\ &\leqq \Pr\left[(EKSK \vee EISK)\right] + \Pr\left[ECB \wedge (\overline{EKSK} \wedge \overline{EISK})\right].\end{aligned}$$

For the event $(\overline{EKSK} \wedge \overline{EISK})$, A_I can't obtain the useful information to output a correct bit b'. A_I has probability 1/2 to guess the correct bit, so $\Pr[ECB \wedge (\overline{EKSK} \wedge \overline{EISK})]$ is still 1/2 on average. Thus, we have:

$$\Pr_{A-I} \leqq \Pr\big[(EKSK \vee EISK)\big] + 1/2,$$
$$Adv_{A-I} \leqq |\Pr_{A-I} - 1/2| = \Pr\big[(EKSK \vee EISK)\big].$$

Because $Adv_{A-I-W} \leqq O(q^2/p)$ and A_I can learn at most 2λ bits of KSK and ISK_{ID^*} by the *Identity secret key leak query* and *Decrypting leak query*, we have

$$Adv_{A-I} \leqq Adv_{A-I-W} \cdot 2^{2\lambda} \leqq O\big((q^2/p) \cdot 2^{2\lambda}\big).$$

By Corollary 1, Adv_{A-I} is negligible if $\lambda < (1 - \varepsilon) \log p$. □

Theorem 2. *In the GBG model, the proposed LR-RCLE-ORA scheme is semantically secure against chosen cipher-text attacks against a revoked user* (A_{II}) *in the security game* $G_{LR\text{-}RCLE\text{-}ORA}$.

Proof. Let A_{II} be a revoked user of the security game $G_{LR\text{-}RCLE\text{-}ORA}$ played with a challenger B. A_{II} may issue various queries to B at most q times in the game. This game consists of four phases as follows:

– *Setup phase*: The phase is the same as that in the proof in Theorem 1. Additionally, B sends the time secret key TSK to A_{II} since it is a revoked user.
– *Query phase*: In this phase, A_{II} can adaptively issue various queries to B at most q times. A_{II} queries are identical to those queries in the proof of Theorem 1. Note that A_{II} knows the personal secret key PSK_{ID} and the identity secret key ISK_{ID} of any user ID. Also, A_{II} may get the time update key $TUK_{ID,s}$ of any user ID at any period T_s, except TUK_{ID^*,s^*} of the target user ID^* at target period T_{s^*}.
– *Challenge phase*: A_{II} sends a target identity ID^*, a target period T_{s^*} and a plaintext pair (msg_0^*, msg_1^*) to B. Here the *Time update key query* (ID^*, T_s^*) must have never been requested by A_{II}. B chooses a unbiased random bit $b \in \{0, 1\}$ and carries out the *Encrypting* algorithm with $(ID^*, T_s^*, (PPK_{ID^*}, IPK_{ID^*}, TUPK_{ID^*,s^*}), msg_b^*)$ to generate C^*, EK^* and $CT^* = E_{EK^*}(msg_b^*)$. Finally, B returns the ciphertext tuple $(ID^*, T_s^*, \theta = (C^*, CT^*))$ to A_{II}.
– *Guess phase*: In this phase, A_{II} must output a bit $b' \in \{0, 1\}$. If $b' = b$, we say that it wins the game $G_{LR\text{-}RCLE\text{-}ORA}$ and its advantage is $|\Pr[b' = b] - 1/2|$.

In the following, let us first evaluate the advantage that A_{II} wins $G_{LR\text{-}RCLE\text{-}ORA}$ without requesting any leak query. Subsequently, the advantage of A_{II} is evaluated when it is allowed to request the *Time update key leak query*.

■ **The advantage of A_{II} without requesting any leak query:** Let Adv_{A-II-W} be the advantage that A_{II} wins the game without requesting the *Time update key leak query*. By the similar evaluations as in the proof of Theorem 1, we have $Adv_{A-II-W} = O(q^2/p)$.

- **The advantage of A_{II} with requesting the Time update key leak query:** By the *Time update key leak query* $(f_{TUKE,j}, h_{TUKE,j}, j)$, A_{II} may derive leaked information $(\Lambda f_{TUKE,j}, \Lambda h_{TUKE,j})$ such that $|f_{TUKE,j}|$, $|h_{TUKE,j}| \leqq \lambda$, where $\Lambda f_{TUKE,j} = f_{TUKE,j}(TSK_{j,1})$ and $\Lambda h_{TUKE,j} = h_{TUKE,j}(TSK_{j,2})$. Indeed, the time secret key TSK has the property in the sense that $TSK = TSK_{0,1} + TSK_{0,2} = TSK_{1,1} + TSK_{1,2} = \cdots = TSK_{i,1} + TSK_{i,2}$. By the refreshing technique, leaked information of $TSK_{i-1,1}/TSK_{i-1,2}$ is independent of that of $TSK_{i,1}/TSK_{i,2}$. Thus, A_{II} may derive at most 2λ bits of TSK.

  Let $\Pr_{A-II}$ and Adv_{A-II} be the probability and advantage, respectively, that A_{II} wins $G_{\text{LR-RCLE-ORA}}$ with requesting the *Time update key leak query*. Two events are defined as below:

  (1) Let $ETSK$ denote the event that A_{II} gets the time secret key TSK by $f_{TUKE,j}$ and $h_{TUKE,j}$. Additionally, $\overline{ETSK}$ is the complement event of $ETSK$.

  (2) Let ECB denote the event that A_{II} outputs a correct b'.

  Hence, the advantage $\Pr_{A-II}$ is $\Pr[ECB]$ and the following inequality holds:

$$\begin{aligned}\Pr_{A-II} &= \Pr[ECB] \\ &= \Pr[ECB \wedge ETSK] + \Pr[ECB \wedge \overline{ETSK}] \\ &\leqq \Pr[ETSK] + \Pr[ECB \wedge \overline{ETSK}].\end{aligned}$$

  For the event $\overline{ETSK}$, A_{II} can't obtain the useful information to output a correct bit b'. A_{II} has probability $1/2$ to guess the correct bit, so $\Pr[ECB \wedge \overline{ETSK}]$ is still $1/2$ on average. Thus, we have

$$\begin{aligned}&\Pr_{A-II} \leqq \Pr[ETSK] + 1/2, \\ &Adv_{A-II} \leqq |\Pr_{A-II} - 1/2| = \Pr[ETSK].\end{aligned}$$

Because $Adv_{A-II-W} \leqq O(q^2/p)$ and A_{II} can learn at most 2λ bits of TSK by the *Time update key leak query*, we have

$$Adv_{A-II} \leqq Adv_{A-II-W} \cdot 2^{2\lambda} \leqq O\big((q^2/p) \cdot 2^{2\lambda}\big).$$

By Corollary 1, Adv_{A-II} is negligible if $\lambda < (1-\varepsilon)\log p$. □

Theorem 3. *In the GBG model, the proposed LR-RCLE-ORA scheme is semantically secure against chosen cipher-text attacks against an honest-but-curious KGC* (A_{III}) *in the security game* $G_{\text{LR-RCLE-ORA}}$.

Proof. Let A_{III} be an honest-but-curious KGC of the security game $G_{\text{LR-RCLE-ORA}}$ played with a challenger B. A_{III} may issue various queries to B at most q times in the game. This game consists of four phases as follows:

– *Setup phase*: The phase is the same as that in the proof in Theorem 1. Additionally, B sends the KGC's secret key KSK and time secret key TSK to A_{III} since it is an honest-but-curious KGC.

– *Query phase*: In this phase, A_{III} can adaptively issue various queries to B at most q times. All queries are identical to those queries in the proof of Theorem 1. Note that A_{III} knows the KGC's secret key KSK and time secret key TSK so that it can get the identity secret key ISK_{ID} and time update key $TUK_{ID,s}$ of any user ID for any period T_s. Also, A_{III} may get the personal secret key PSK_{ID} of any user ID, except PSK_{ID^*} of the target user ID^*.
– *Challenge phase*: A_{III} sends a target identity ID^*, a target period T_{s^*} and a plaintext pair (msg_0^*, msg_1^*) to B. Here both the *Personal secret key corrupt query* (ID^*) and the *Public key replace query* $(ID^*, T_{s*}, (PPK'_{ID^*}, IPK'_{ID^*}, TUPK'_{ID^*,s^*}))$ must have never been requested by A_{III}. B chooses an unbiased random bit $b \in \{0, 1\}$ and carries out the *Encrypting* algorithm with $(ID^*, T_s^*, (PPK_{ID^*}, IPK_{ID^*}, TUPK_{ID^*,s^*}), msg_b^*)$ to generate C^*, EK^* and $CT^* = E_{EK^*}(msg_b^*)$. Finally, B returns the ciphertext tuple $(ID^*, T_s^*, \theta = (C^*, CT^*))$ to A_{III}.
– *Guess phase*: In this phase, A_{III} must output a bit $b' \in \{0, 1\}$. If $b' = b$, we say that it wins the game $G_{LR\text{-}RCLE\text{-}ORA}$ and its advantage is $|\Pr[b' = b] - 1/2|$.

In the following, let us first evaluate the advantage that A_{III} wins $G_{LR\text{-}RCLE\text{-}ORA}$ without requesting any leak query. Subsequently, the advantage of A_{III} is evaluated when it is allowed to request the *Decrypting leak query* $(ID, T_s, f_{DEC,k}, h_{DEC,k}, k)$.

■ **The advantage of A_{III} without requesting any leak query:** Let $Adv_{A-III-W}$ be the advantage that A_{III} wins the game without requesting the *Decrypting leak query*. By the similar evaluations as in the proof of Theorem 1, we have $Adv_{A-III-W} = O(q^2/p)$.
■ **The advantage of A_{III} with requesting the Decrypting leak query:** By the *Decrypting leak query* $(ID, T_s, f_{DEC,k}, h_{DEC,k}, k)$, A_{III} may derive leaked information $(\Lambda f_{DEC,k}, \Lambda h_{DEC,k})$ such that $|f_{DEC,k}|, |h_{DEC,k}| \leqq \lambda$, where $\Lambda f_{DEC,k} = f_{DEC,k}(PSK_{ID,k,1})$ and $\Lambda h_{DEC,k} = h_{DEC,k}(PSK_{ID,k,2})$. Note that A_{III} may get the personal secret key PSK_{ID} of any user ID, except PSK_{ID^*} of the target user ID^*. For leakage resilient property, A_{III} can obtain leaked information of the target user's $PSK_{ID^*} = (PSK_{ID^*,k,1}, PSK_{ID^*,k,2})$ used in the *Decrypting* algorithm's k-th round of the target user. Indeed, the personal secret key PSK_{ID} has the property in the sense that $PSK_{ID^*} = PSK_{ID^*,0,1} + PSK_{ID^*,0,2} = PSK_{ID^*,1,1} + PSK_{ID^*,1,2} = \cdots = PSK_{ID^*,k,1} + PSK_{ID^*,k,2}$. By the refreshing technique, leaked information of $PSK_{ID^*,k-1,1}/PSK_{ID^*,k-1,2}$ is independent of that of $PSK_{ID^*,k,1}/PSK_{ID^*,k,2}$. Thus, A_{III} may derive at most 2λ bits of PSK_{ID^*}.

Let $\Pr_{A-III}$ and Adv_{A-III} be the probability and advantage, respectively, that A_{III} wins $G_{LR\text{-}RCLE\text{-}ORA}$ with requesting the *Decrypting leak query*. Two events are defined as below:
(1) Let $EPSK$ denote the event that A_{III} gets the personal secret key PSK_{ID^*} by $f_{DEC,k}$ and $h_{DEC,k}$. Additionally, $\overline{EPSK}$ is the complement event of $EPSK$.
(2) Let ECB denote the event that A_{III} outputs a correct b'.

Table 1
Computational time (in millisecond) of two time-consuming operations.

Notation	Operation	Computational time
T_p	a bilinear pairing $\hat{e} : G_1 \times G_1 \rightarrow G_2$	7.84 ms
T_{me}	a scalar multiplication on an additive group G_1 or an exponentiation on a multiplicative group G_2	0.48 ms

Hence, the advantage $\Pr_{A-III}$ is $\Pr[ECB]$ and the following inequality holds.

$$\begin{aligned}\Pr_{A-III} &= \Pr[ECB] \\ &= \Pr[ECB \wedge EPSK] + \Pr[ECB \wedge \overline{EPSK}] \\ &\leqq \Pr[EPSK] + \Pr[ECB \wedge \overline{EPSK}].\end{aligned}$$

For the event $\overline{EPSK}$, A_{III} can't obtain the useful information to output a correct bit b'. A_{III} has probability 1/2 to guess the correct bit, so $\Pr[ECB \wedge \overline{EPSK}]$ is still 1/2 on average. Thus, we have

$$\begin{aligned}&\Pr_{A-III} \leqq \Pr[EPSK] + 1/2, \\ &Adv_{A-III} \leqq |\Pr_{A-III} - 1/2| = \Pr[EPSK].\end{aligned}$$

Because $Adv_{A-III-W} \leqq O(q^2/p)$ and A_{III} can learn at most 2λ bits of PSK by the *Decrypting leak query*, we have

$$Adv_{A-III} \leqq Adv_{A-III-W} \cdot 2^{2\lambda} \leqq O\big((q^2/p) \cdot 2^{2\lambda}\big).$$

By Corollary 1, Adv_{A-III} is negligible if $\lambda < (1 - \varepsilon) \log p$. □

6. Comparisons

Here, let's first present the computation notations of several operations in bilinear groups. By employing the simulation experiences in Li *et al.* (2021), Table 1 lists two kinds of time-consuming operations and their computational time (in milliseconds). The environment of simulation experiences is a platform with the Intel Core i7-8550U CPU 1.80 GHz processor. The selection of security parameters are F_p, G_1 and G_2. Here, F_p is a finite field which consists of the set of integers $\{0, 1, \ldots, p - 1\}$, where p is a 256-bit prime number. And, G_1 and G_2 are groups with 224-bit prime order over the finite field F_p. It is worth mentioning that the computation of a one-way hash function, an addition on an additive group G_1 and a multiplication on a multiplicative group G_2 are omitted because their computational costs are small and negligible.

Table 2 lists the comparisons of our LR-RCLE-ORA scheme with several RCLE and LR-CLE schemes (Tsai and Tseng, 2015; Xiong *et al.*, 2013; Wu *et al.*, 2018) in terms of

Table 2
Comparisons between our scheme and the previously proposed schemes.

	Tsai and Tseng's RCLE scheme (Tsai and Tseng, 2015)	Xiong *et al.*'s LR-CLE scheme (Xiong *et al.*, 2013)	Wu *et al.*'s LR-CLE scheme (Wu *et al.*, 2018)	Our LR-RCLE-ORA scheme
Encryption cost	$3T_{me} + T_p$ (9.28 ms)	$6T_{me}$ (2.88 ms)	$4T_{me} + T_p$ (9.76 ms)	$4T_{me} + 2T_p$ (17.6 ms)
Decryption cost	$2T_{me} + T_p$ (8.8 ms)	$4T_p$ (31.36 ms)	$4T_{me} + 4T_p$ (33.28 ms)	$4T_{me} + 5T_p$ (41.12 ms)
Security proof model	Random oracle model	Standard model (Dual system)	GBG model	GBG model
Revocation property	Yes	No	No	Yes
Outsourced revocation	No	No	No	Yes
Resisting side-channel attacks	No	Yes	Yes	Yes
Leakage resilience model	No	Bounded	Continual	Continual

encryption cost (time), decryption cost (time), security proof model, revocation property, outsourced revocation, resisting side-channel attacks and leakage resilience model. Note that a user's private key in Xiong *et al.*'s LR-CLE scheme (2013) is a vector with $n \geqq 2$ elements (here, let $n = 2$). As compared to the bounded leakage property of Xiong *et al.*'s LR-CLE scheme (2013), Wu *et al.*'s scheme and ours possess continual leakage property and are practical for applications. To resist side-channel attacks, our scheme requires some extra computation costs than the RCLE scheme in Tsai and Tseng (2015). As compared with the LR-CLE scheme (Wu *et al.*, 2018), our scheme requires one T_p for encryption cost and decryption cost, respectively, but our scheme offers outsourced revocation functionality to reduce the computational burden of the KGC for generating all non-revoked users' time update keys. By Table 2, even though the computational cost of our scheme is worse than the other schemes, our scheme possesses four complete properties, namely, revocation property, outsourced revocation, resisting side-channel attacks and continual leakage property. We emphasize that our scheme is the first LR-RCLE-ORA scheme resisting side-channel attacks while possessing continual leakage property.

7. Conclusions

In this paper, the first leakage-resilient revocable certificateless encryption scheme with an outsourced revocation authority (LR-RCLE-ORA) was proposed. As compared to previous RCLE and LR-CLE schemes, our scheme possesses several merits. (1) It can resist side-channel attacks and has leakage resilience properties. (2) The revocation functionality is outsourced to the ORA to alleviate the computational load of the KGC. (3) It permits adversaries to continually extract partial ingredients of secret keys and offers the overall unbounded leakage property. By extending the adversary models of RCLE and LR-CLE schemes, a new adversary model was defined while three kinds of leak queries are added, namely, *Identity secret key leak query*, *Time update key leak query* and *decrypting leak query*. In the GBG model, the security of the proposed scheme is shown to be semantically secure against chosen cipher-text attacks for three kinds of adversaries, namely, outsider, revoked user and honest-but-curious KGC.

Funding

This research was partially supported by Ministry of Science and Technology, Taiwan, under contract no. MOST110-2221-E-018-006-MY2, MOST110-2221-E-018-007-MY2 and MOST110-2222-E-019-001-MY2.

References

Abdalla, M., Belaid, S., Fouque, P. (2013). Leakage-resilient symmetric encryption via re-keying. In: *CHES'13*, *LNCS*, Vol. 8086, pp. 471–488.

Akavia, A., Goldwasser, S., Vaikuntanathan, V. (2009). Simultaneous hardcore bits and cryptography against memory attacks. In: *TCC'09*, *LNCS*, Vol. 5444, pp. 474–495.

Al-Riyami, S.S., Paterson, K.G. (2003). Certificateless public key cryptography. In: *ASIACRYPT'03*, *LNCS*, Vol. 2894, pp. 452–473.

Alwen, J., Dodis, Y., Wichs, D. (2009). Leakage-resilient public-key cryptography in the bounded-retrieval model. In: *CRYPTO'09*, *LNCS*, Vol. 5677, pp. 36–54.

Boneh, D., Franklin, M. (2001). Identity-based encryption from the Weil pairing. In: *CRYPTO'01*, *LNCS*, Vol. 2139, pp. 213–229.

Boneh, D., Boyen, X., Goh, E.J. (2005). Hierarchical identity-based encryption with constant size ciphertext. In: *EUROCRYPT*, *LNCS*, Vol. 3494, pp. 440–456.

Brakerski, Z., Kalai, Y.T., Katz, J., Vaikuntanathan, V. (2010). Cryptography resilient to continual memory leakage. In: *51st Annual IEEE Symposium on Foundations of Computer Science*. IEEE Press, pp. 501–510.

Bronchain, O., Momin, C., Peters, T., Standaert, F. (2021). Improved leakage-resistant authenticated encryption based on hardware AES coprocessors. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(3), 641–676.

Brumley, D., Boneh, D. (2005). Remote timing attacks are practical. *Computer Networks*, 48(5), 701–716.

Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A. (2008). Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1), 97–139.

Du, H., Wen, Q., Zhang, S. (2018). A provably-secure outsourced revocable certificateless signature scheme without bilinear pairings. *IEEE Access*, 6, 73846–73855.

Galindo, D., Virek, S. (2013). A practical leakage-resilient signature scheme in the generic group model. In: *SAC'12*, *LNCS*, Vol. 7707, pp. 50–65.

Galindo, D., Grobschadl, J., Liu, Z., Vadnala, P.K., Vivek, S. (2016). Implementation of a leakage-resilient ElGamal key encapsulation mechanism. *Journal of Cryptographic Engineering*, 6(3), 229–238.

Hazay, C., López-Alt, A., Wee, H., Wichs, D. (2013). Leakage-resilient cryptography from minimal assumptions. In: *EUROCRYPT'13*, *LNCS*, Vol. 7881, pp. 160–176.

Housley, R., Polk, W., Ford, W., Solo, D. (2002). *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. IETF, RFC 3280.

Hsieh, T.-C., Tseng, Y.-M., Huang, S.-S. (2020). A leakage-resilient certificateless authenticated key exchange protocol withstanding side-channel attacks. *IEEE Access*, 8, 121795–121810.

Hung, Y.-H., Tseng, Y.-M., Huang, S.S. (2016). A revocable certificateless short signature scheme and its authentication application. *Informatica*, 27(3), 549–572.

Katz, J., Vaikuntanathan, V. (2009). Signature schemes with bounded leakage resilience. In: *ASIACRYPT'09*, *LNCS*, Vol. 5912, pp. 703–720.

Kiltz, E., Pietrzak, K. (2010). Leakage resilient elgamal encryption. In: *ASIACRYPT'10*, *LNCS*, Vol. 6477, pp. 595–612.

Kocher, P.C. (1996). Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: *CRYPTO'96*, *LNCS*, Vol. 1163, pp. 104–113.

Kocher, P., Jaffe, J., Jun, B. (1999). Differential power analysis. In: *CRYPTO'99*, *LNCS*, Vol. 1666, pp. 388–397.

Li, Y., Cheng, Q., Liu, X., Li, X. (2021). A secure anonymous identity-based scheme in new authentication architecture for mobile edge computing. *IEEE Systems Journal*, 15(1), 935–946.

Li, J., Guo, Y., Yu, Q., Lu, Y., Zhang, Y. (2016). Provably secure identity based encryption resilient to post-challenge continuous auxiliary input leakage. *Security and Communication Network*, 9(10), 1016–1024.

Li, S., Zhang, F., Sun, Y., Shen, L. (2013). Efficient leakage-resilient public key encryption from DDH assumption. *Cluster Computing*, 16(4), 797–806.

Liu, S., Weng, J., Zhao, Y. (2013). Efficient public key cryptosystem resilient to key leakage chosen ciphertext attacks. In: *CTRSA'13*, *LNCS*, Vol. 7779, pp. 84–100.

Naor, M., Segev, G. (2009). Public-key cryptosystems resilient to key leakage. In: *CRYPTO'09*, *LNCS*, Vol. 5677, pp. 18–35.

Naor, M., Segev, G. (2012). Public-key cryptosystems resilient to key leakage. *SIAM Journal on Computing*, 41(4), 772–814.

Scott, M. (2011). On the efficient implementation of pairing-based protocols. In: *Cryptography and Coding*, *LNCS*, Vol. 7089, pp. 296–308.

Shen, L., Zhang, F., Sun, Y. (2014). Efficient revocable certificateless encryption secure in the standard model. *Computer Journal*, 57(4), 592–601.

Tsai, T.-T., Tseng, Y.-M. (2015). Revocable certificateless public key encryption. *IEEE Systems Journal*, 9(3), 824–833.

Tsai, T.-T., Tseng, Y.-M., Huang, S.-S. (2015). Efficient revocable certificateless public key encryption with a delegated revocation authority. *Security and Communication Networks*, 8(18), 3713–3725.

Tsai, T.-T., Chuang, Y.-H., Tseng, Y.-M., Huang, S.-S., Hung, Y.-H. (2021). A leakage-resilient ID-based authenticated key exchange protocol with a revocation mechanism. *IEEE Access*, 9, 128633–128647.

Tseng, Y.-M., Tsai, T.-T. (2012). Efficient revocable ID-based encryption with a public channel. *Computer Journal*, 55(4), 475–486.

Tseng, Y.-M., Wu, J.-D., Huang, S.-S., Tsai, T.-T. (2020). Leakage-resilient outsourced revocable certificateless signature with a cloud revocation server. *Information Technology and Control*, 49(4), 464–481.

Unterstein, F., Schink, M., Schamberger, T., Tebelmann, L., Ilg, M., Heyszl, J. (2020). Retrofitting leakage resilient authenticated encryption to microcontrollers. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(4), 365–388.

Wu, J.-D., Tseng, Y.-M., Huang, S.-S., Chou, W.-C. (2018). Leakage-resilient certificateless key encapsulation scheme. *Informatica*, 29(1), 125–155.

Wu, J.-D., Tseng, Y.-M., Huang, S.-S. (2019). An identity-based authenticated key exchange protocol resilient to continuous key leakage. *IEEE Systems Journal*, 13(4), 3968–3979.

Wu, J.-D., Tseng, Y.-M., Huang, S.-S., Tsai, T.-T. (2020). Leakage-resilient revocable identity-based signature with cloud revocation authority. *Informatica*, 31(3), 597–620.

Xiong, H., Yuen, T.-H., Zhang, C., Yiu, S.-M., He, Y.-J. (2013). Leakage-resilient certificateless public key encryption. In: *The first ACM workshop on Asia Public-Key Cryptography*. ACM Press, pp. 13–22.

Yuen, T.-H., Chow, S.S.M., Zhang, Y., Yiu, S.-M. (2012). Identity-based encryption resilient to continual auxiliary leakage. In: *EUROCRYPT'12*, *LNCS*, Vol. 7237, pp. 117–134.

Zhou, Y., Yang, B., Zhang, W. (2016). Provably secure and efficient leakage-resilient certificateless signcryption scheme without bilinear pairing. *Discrete Applied Mathematics*, 204, 185–202.

Y.-M. Tseng is currently the vice president and a professor in the Department of Mathematics, National Changhua University of Education, Taiwan. He is a member of IEEE Computer Society, IEEE Communications Society and the Chinese Cryptology and Information Security Association (CCISA). He has published over one hundred scientific journal papers on various research areas of cryptography, security and computer network. His research interests include cryptography, network security, computer network and leakage-resilient cryptography. He serves as an editor of several international journals.

S.-S. Huang is currently a professor in the Department of Mathematics, National Changhua University of Education, Taiwan. His research interests include number theory, cryptography, and leakage-resilient cryptography. He received his PhD from the University of Illinois at Urbana-Champaign in 1997 under the supervision of Professor Bruce C. Berndt.

T.-T. Tsai is currently an assistant professor in the Department of Computer Science and Engineering, National Taiwan Ocean University, Taiwan. His research interests include applied cryptography, pairing-based cryptography and leakage-resilient cryptography. He received the PhD degree from the Department of Mathematics, National Changhua University of Education, Taiwan, in 2014, under the supervision of Professor Yuh-Min Tseng.

Y.-H. Chuang received the PhD degree from the Department of Computer Science and Engineering, National Taiwan University, Taiwan, in 2020. His research interests include information security, cryptography and leakage-resilient cryptography.

Y.-H. Hung received the PhD degree from the Department of Mathematics, National Changhua University of Education, Taiwan, in 2017 under the supervision of Professor Yuh-Min Tseng. His research interests include applied cryptography and pairing-based cryptography.