INFORMATICA, 2018, Vol. 29, No. 1, 125–155

DOI: http://dx.doi.org/10.15388/Informatica.2018.161

Leakage-Resilient Certificateless Key Encapsulation Scheme

Jui-Di WU, Yuh-Min TSENG[4]*, Sen-Shan HUANG, Wei-Chieh CHOU
Department of Mathematics, National Changhua University of Education Jin-De Campus
Chang-Hua City 500, Taiwan
e-mail: ymtseng@cc.ncue.edu.tw

Received: June 2017; accepted: January 2018

Abstract. The previous adversary models of public key cryptography usually have a nature assumption that permanent/temporary secret (private) keys must be kept safely and internal secret states are not leaked to an adversary. However, in practice, it is difficult to keep away from all possible kinds of leakage on these secret data due to a new kind of threat, called “side-channel attacks”. By side-channel attacks, an adversary could obtain partial information of these secret data so that some existing adversary models could be insufficient. Indeed, the study of leakage-resilient cryptography resistant to side-channel attacks has received significant attention recently. Up to date, no work has been done on the design of leakage-resilient certificateless key encapsulation (LR-CL-KE) or public key encryption (LR-CL-PKE) schemes under the continual leakage model. In this article, we propose the *first* LR-CL-KE scheme under the continual leakage model. Moreover, in the generic bilinear group (GBG) model, we formally prove that the proposed LR-CL-KE scheme is semantically secure against chosen ciphertext attacks for both Type I and Type II adversaries.

Key words: certificateless encryption, continual leakage model, side-channel attacks, leakage resilience, generic bilinear group model.

1. Introduction

To simplify public key management and remove the need of certificates required in the traditional public key cryptography, Shamir (1984) presented the notion of identity (ID)-based public key cryptography (ID-PKC). However, ID-PKC encounters the key escrow problem in the sense that the private key generator (PKG) knows all users’ private keys so that the PKG may decrypt all the ciphertexts or sign the messages on behalf of all the users. In order to solve the key escrow problem in ID-PKC, Al-Riyami and Paterson (2003) proposed the certificateless public key cryptography (CL-PKC), in which there are two players, namely, a key generation centre (KGC) and users. The KGC represents a trusted third party and is responsible to generate each user’s initial key. Each user’s full private key consists of two components, namely, the initial key generated by the KGC and a secret key chosen by the user. Meanwhile, in accordance with the secret key, a user can compute her/his corresponding public key. Obviously, the KGC can’t obtain

[*] Corresponding author.

a user's full private key due to the lack of the user's self-chosen secret key. Therefore, CL-PKC overcomes the key escrow problem and retains the advantage of eliminating certificates in ID-PKC. Indeed, the study on CL-PKC has received great attention from researchers and a large number of certificateless cryptographic schemes have been proposed such as certificateless public-key encryption (CL-PKE) (Libert and Quisquater, 2006; Hwang *et al.*, 2008; Tsai *et al.*, 2015; Tsai and Tseng, 2015; Hung *et al.*, 2017) and certificateless signature (CLS) (Huang *et al.*, 2007; Hu *et al.*, 2007; Hung *et al.*, 2015, 2016). The mentioned certificateless encryption/signature schemes above were implemented by employing bilinear pairing groups. However, the operations in bilinear pairing groups are more time-consuming than the exponentiation operator in RSA groups. Recently, several RSA-based certificateless encryption/signature schemes (Zhang and Mao, 2012; Sharma *et al.*, 2016; Lin *et al.*, 2017) were proposed to improve computation performance of pairing-based certificateless encryption/signature schemes.

Nevertheless, the previous adversary models of traditional, ID-based and certificateless public key cryptographies usually have a nature assumption that permanent/temporary secret (private) keys must be kept safely and internal secret states are not leaked to an adversary. However, in practice, it is difficult to keep away from all possible kinds of leakage on these secret data due to a new kind of threat, called "side-channel attacks", such as timing attacks (Kocher, 1996; Brumley and Boneh, 2005), power analysis (Kocher *et al.*, 1999) and fault attacks (Boneh *et al.*, 1997; Biham *et al.*, 2008). By side-channel attacks, an adversary could obtain partial information of these secret data so that some existing adversary models could be insufficient. More precisely, if a cryptographic scheme was proven secure in an adversary model without addressing side-channel attacks, the cryptographic scheme still could be broken in an environment where an adversary may obtain partial information of secret data. Therefore, the study of leakage-resilient cryptography (LRC) resisting to side-channel attacks has received significant attention recently.

The basic concept of LRC is that a cryptographic scheme remains secure when partial information of the secret data involved in the scheme is visible to the adversary. In order to represent the leakage resilience of cryptographic schemes, an adversary model of LRC must define the adversary's capabilities of obtaining leakage information. A cryptographic scheme typically includes several calculation rounds. For each calculation round, the adversary has a leakage function f on the secret data τ and may obtain the leakage information $f(\tau)$. Also, the output length of f is limited to λ bits. Namely, the adversary can obtain at most λ bits of leakage information for each calculation round. However, the full secret (private) key would be exposed to the adversary if the total leakage information of a cryptographic scheme is unbounded. In such a case, it will compromise the security of the cryptographic scheme. Hence, several leakage-resilient cryptographic schemes (Akavia *et al.*, 2009; Alwen *et al.*, 2009; Katz and Vaikuntanathan, 2009) make a restriction that the total leakage information must be bounded which is called the *bounded leakage model*. However, this restriction is impractical. Indeed, the *continual leakage model* is the most accredited model for leakage-invocated ability of an adversary, which provides the *overall unbounded leakage* property (Brakerski *et al.*, 2010; Dodis and Haralambiev, 2010; Galindo and Virek, 2013; Wu *et al.*, 2016). The properties of the continual leakage model will be reviewed in Section 3.

1.1. *Related Work*

Akavia *et al.* (2009) presented the first security model of leakage-resilient public key encryption (LR-PKE) in the bounded leakage model. In their security model, an adversary can select arbitrary leakage functions of the secret (private) keys and obtains the outputs of these functions. They also proposed a concrete LR-PKE scheme, which is the first leakage-resilient chosen plaintext attack (LR-CPA) secure scheme. Naor and Segev (2009, 2012) extended Akavia *et al.*'s security model of LR-PKE scheme to present the settings of both the leakage-resilient chosen ciphertext attacks (LR-CCA1) and the adaptive leakage resilient chosen ciphertext attacks (LR-CCA2). Meanwhile, Naor and Segev also presented a generic construction of LR-PKE scheme from the universal hash proof system. Liu *et al.* (2013) and Li *et al.* (2013), respectively, proposed efficient LR-PKE schemes which have less computational cost than Naor and Segev's scheme (2009, 2012). The schemes mentioned above are all secure in bounded leakage model, but not under the continual leakage model. Moreover, Kiltz and Pietrzak (2010) proposed a leakage-resilient public key encryption under the continual leakage model using the generic bilinear group (GBG) model (Boneh *et al.*, 2005). The properties of the GBG model will be presented in Section 2. Based on the GBG model, Galindo *et al.* (2016) also presented and implemented a new ElGamal-like leakage-resilient key encapsulation (LR-KE) scheme under the continual leakage model. All the LR-PKE schemes mentioned above are based on traditional public key settings.

In ID-based public key settings, Brakerski *et al.* (2010) proposed the first leakage-resilient ID-based encryption (LR-IBE) scheme under the continual leakage model. Afterwards, Yuen *et al.* (2012) proposed an improved LR-IBE scheme to achieve better performance. Their scheme allows an adversary to learn partial information of both the system secret key in the key extract phase and the user's private key in the decryption phase. Recently, Li *et al.* (2016) presented a new LR-IBE scheme under composite order groups. By the post-challenge continuous auxiliary input, their scheme is secure against adaptive chosen plaintext attacks under three static assumptions in the standard model.

Indeed, there exists little work on leakage-resilient certificateless cryptographic schemes. Xiong *et al.* (2013) proposed the first leakage-resilient certificateless public key encryption (LR-CL-PKE) scheme, which is secure against Type I (outsider) and Type II (honest-but-curious KGC) adversaries. Xiong *et al.*'s scheme possesses the security against LR-CPA and LR-CCA1 attacks, but not against LR-CCA2 attacks. Zhou *et al.* (2016) improved Xiong *et al.*'s scheme to propose a LR-CCA2 secure leakage-resilient certificateless signcryption scheme based on bilinear pairings. However, both Xiong *et al.*'s and Zhou *et al.*'s schemes are secure under the bounded leakage model, but not under the continual leakage model.

1.2. *Contributions*

Up to date, no existing leakage-resilient certificateless public key encryption (LR-CL-PKE) or leakage-resilient certificateless key encapsulation (LR-CL-KE) schemes are secure under the continual leakage model. In this article, we will propose the first LR-CL-KE scheme under the continual leakage model. We first define the adversary model of LR-CL-KE schemes under the continual leakage model. The adversary model is extended from the adversary model of CL-PKE schemes defined in Hwang *et al.* (2008), Tsai *et al.* (2015), Tsai and Tseng (2015). The adversary model also consists of two types of adversaries, namely, Type I adversary (outsider) and Type II adversary (honest-but-curious KGC). By adding the leak queries, the new adversary model of LR-CL-KE schemes allows to leak partial information of the system secret key in the initial key extract phase and leak the partial information of the user's private key in the decrypt phase. The point is that the adversary model provides the *overall unbounded leakage* property (Galindo and Virek, 2013; Wu *et al.*, 2016) under the continual leakage model. In the generic bilinear group (GBG) model (Boneh *et al.*, 2005), we formally prove that the proposed LR-CL-KE scheme is semantically secure against chosen ciphertext attacks for both Type I and Type II adversaries. Finally, the performance analysis is given to demonstrate the comparison of the proposed LR-CL-KE scheme and the related schemes.

1.3. *Organization*

The remainder of the article is organized as follows. Preliminaries are given in Section 2. In Section 3, we present the framework and security notions of LR-CL-KE schemes. Then a concrete LR-CL-KE scheme is proposed in Section 4. We analyse the security of the proposed LR-CL-KE scheme in Section 5. Section 6 demonstrates performance comparisons. Conclusions and future work are given in Section 7.

2. Preliminaries

The notions of bilinear groups (Boneh and Franklin, 2001; Waters, 2005; Scott, 2011), the properties of the generic bilinear group model (Galindo and Virek, 2013; Boneh *et al.*, 2005; Wu *et al.*, 2016) and entropy are briefly introduced here.

2.1. *Bilinear Pairings*

Let G and G_T be two cyclic multiplicative groups of large prime order p. Let g be a generator of the group G. An admissible bilinear pairing is a map e: $G \times G \to G_T$ and satisfies the following conditions:

(1) *Bilinearity*: for all $x, y \in Z_p^*$, $e(g^x, g^y) = e(g, g)^{xy}$.
(2) *Non-degeneracy*: for some $g \in G$, $e(g, g) \neq 1$.
(3) *Computability*: for all $g_1, g_2 \in G$, $e(g_1, g_2)$ can be efficiently computed.

In addition, G is a bilinear group while G_T is called the target group of the admissible bilinear map e. A reader can refer to previous literatures such as Boneh and Franklin (2001), Waters (2005), Scott (2011) for more complete descriptions about bilinear groups and admissible bilinear map.

2.2. *Generic Bilinear Group Model*

The notions of the generic group model were first introduced by Shoup (1997), which is viewed as an adversary model for cryptographic schemes. In the generic group model, an adversary can issue a group oracle (query) to a challenger for executing the group operation (Maurer and Wolf, 1998). The group operation takes as input two group elements and outputs third group element. For example, in a multiplicative group, the group operation is multiplication which multiplies two group elements together to obtain third group element. Namely, the group oracle allows an adversary to have access to a randomly chosen encoding (element) of a group controlled by the challenger. Meanwhile, if the used group allows the other pairing operation such as bilinear pairing, an additional oracle must be provided. One of the main usages of the generic group model is to analyse *computational hardness assumptions* such as the discrete logarithm problem. It is said to solve the computational hardness assumption if an adversary can efficiently find a collision element of a group operation.

Boneh *et al.* (2005) extended the generic group model above to present the generic bilinear group (GBG) model. In the GBG model, there exist two multiplicative cyclic groups G and G_T with three operations, namely, group operations of G and G_T, respectively, and a bilinear pairing operation from $G \times G$ into G_T. The elements of G and G_T are respectively encoded by two random injective maps $\varepsilon : Z_p \to \phi$ and $\varepsilon_T : Z_p \to \phi_T$, where both ϕ and ϕ_T are bit strings such that $|\phi \cap \phi_T| = 0$ and $|\phi| = |\phi_T| = p$. Meanwhile, in the GBG model, two queries (oracles) Q_G and Q_T are provided to perform the associated group multiplication operations in G and G_T while a query Q_P is used to perform the evaluation of the bilinear map e. For any $x, y \in Z_p^*$, three queries have the following properties respectively.

- $Q_G(\varepsilon(x), \varepsilon(y)) \to \varepsilon(x + y \bmod p)$.
- $Q_T(\varepsilon_T(x), \varepsilon_T(y)) \to \varepsilon_T(x + y \bmod p)$.
- $Q_P(\varepsilon(x), \varepsilon(y)) \to \varepsilon_T(xy \bmod p)$.

Note that $\varepsilon(1) = g$ and $e(g, g) = \varepsilon_T(1) = g_T$, where g and g_T are generators of the groups G and G_T, respectively.

2.3. *Entropy*

Entropy is a number measure of possible states (or microstates) of a system. In addition, the interpretation of entropy in statistics is viewed as the measure of uncertainty. We assume that X is a finite random variable and Pr is the associated probability distribution. The worst-case predictability of a random variable is measured by using min-entropy. Two kinds of min-entropies are defined as follows:

(1) $H_\infty(X) = -\log_2(\max_x \Pr[X = x])$ denotes the min-entropy of a finite random variable X.

(2) $\widetilde{H}_\infty(X|Z) = -log_2(E_{z \leftarrow Z}[\max_x \Pr[X = x|Z = z]])$ denotes the average conditional min-entropy of a random variable X under a correlated random variable Z.

Under some condition on the leakage information, Dodis *et al.* (2008) presented the min-entropy of a finite random variable X by the following Lemma 1.

Lemma 1. *Assume that $f : X \to 0, 1^{\lambda}$ is a leakage function on a secret random variable X and $f(X)$ denotes the leakage information while the output length of f is limited to λ bits. We have $\widetilde{H}_{\infty}(X|f(X)) \geqslant H_{\infty}(X) - \lambda$.*

In addition, Galindo and Virek (2013) proved Lemma 2 below to demonstrate the probability distribution of a polynomial under the leakage information. Their result is an extension of the Schwartz–Zippel lemma (Zippel, 1979; Schwartz, 1980).

Lemma 2. *Assume that $F \in Z_p[X_1, X_2, \ldots, X_n]$ denotes a non-zero polynomial of total degree at most d. Let P_i (for $i = 1, 2, \ldots, n$) be probability distributions on Z_p such that $H_{\infty}(P_i) \geqslant \log(p) - \lambda$ and $0 \leqslant \lambda \leqslant \log(p)$. If $x_i \xleftarrow{P_i} Z_p$ (for $i = 1, 2, \ldots, n$) are independent, we have the probability $Pr[F(x_1, x_2, \ldots, x_n) = 0] \leqslant \frac{d}{p}2^{\lambda}$.*

The following result follows directly from Lemma 2.

Corollary 1. *The probability $Pr[F(x_1, x_2, \ldots, x_n) = 0]$ is negligible if $\lambda < \log p - \omega(\log(\log(p)))$.*

3. Framework and Security Notions

In this section, we define the framework (syntax) and security notions (adversary model) of leakage-resilient certificateless key encapsulation (LR-CL-KE) schemes under the continual leakage model. Here, we first introduce the properties of the continual leakage model as follows:

- *Only computation leakage*: This property means that only permanent/temporary secret (private) keys accessed and involved in a current calculation round could be leaked to a side-channel adversary.
- *Bounded leakage of single observation*: The length of leakage information in a single calculation round (observation) is limited to some λ bits. This property indicates that the leakage information of each calculation round is bounded to some fraction of secret information.
- *Independent leakage*: The leakage information of all the calculation rounds is independent with each other.
- *Overall unbounded leakage*: This property means that the total amount of leakage information is unbounded. In such a case, secret (private) keys must be updated (refreshed) before/after each calculation round.

In order to achieve the overall unbounded leakage, a continual leakage model must possess the *stateful* property (Kiltz and Pietrzak, 2010). Firstly, each secret (private) key must be divided into two parts and stored in different parts of the memory. If secret (private) keys are updated before (or after) executing the calculation round in a cryptographic algorithm while the associated public key remains fixed, we say that the cryptographic scheme under continual leakage model provides *stateful* property.

3.1. *Framework of LR-CL-KE Scheme*

Here, we present the framework of LR-CL-KE scheme under the continual leakage model.

Definition 1. A LR-CL-KE scheme consists of seven algorithms:

- *Setup:* Taking a security parameter as input, the key generation centre (KGC) runs this algorithm to generate the first system secret key $(SK_{0,1}, SK_{0,2})$ and the public parameters PP. The KGC then publishes PP and keeps $(SK_{0,1}, SK_{0,2})$ in secret. The KGC also selects a symmetric cryptosystem with encryption function $E()$ and decryption function $D()$.
- *Initial key extract*: For the i-th user with identity ID, the KGC uses this algorithm to generate the first initial key (DID_0, QID) of the user. This algorithm consists of two sub-algorithms *Extract-1* and *Extract-2* defined below, in which the current system secret key $(SK_{i-1,1}, SK_{i-1,2})$ is used and is updated to $(SK_{i,1}, SK_{i,2})$.
  - *Extract-1*: Given a random number γ, $SK_{i-1,1}$ and the user's identity ID, this sub-algorithm generates QID and temporary information TI_{IE}, and updates $SK_{i-1,1}$ to $SK_{i,1}$.
  - *Extract-2*: Given TI_{IE}, and $SK_{i-1,2}$, this sub-algorithm generates DID_0 and updates $SK_{i-1,2}$ to $SK_{i,2}$.

  The KGC then sends the first initial key (DID_0, QID) to the user.
- *Set secret value*: This algorithm is performed by a user with identity ID to generate the user's secret key SID_0 and the partial public key RID.
- *Set private key*: This algorithm is performed by a user with identity ID. This algorithm takes the user's first initial key (DID_0, QID) and secret key SID_0 as input to set the user's private key $((DID_{0,1}, DID_{0,2}), (SID_{0,1}, SID_{0,2}))$.
- *Set public key*: This algorithm is performed by a user with identity ID. This algorithm takes the initial key (DID_0, QID) and the partial public key RID as input, and outputs the user's public key $PID = (QID, RID)$.
- *Encrypt*: Given a plain-message msg and the public key $PID = (QID, RID)$ of a receiver with identity ID, this algorithm first generates a random value C and the associated encryption key K, and then generates $CT = E_K(msg)$ by using the encryption function $E()$ of a symmetric cryptosystem. Finally, (C, CT) is sent to the receiver.
- *Decrypt*: This algorithm consists of two sub-algorithms *Decrypt-1* and *Decrypt-2*, run by a receiver. For the j-th *Decrypt* round, the user with identity ID adopts her/his current private key $((DID_{j-1,1}, DID_{j-1,2}), (SID_{j-1,1}, SID_{j-1,2}))$ to decrypt the ciphertext (C, CT) by performing two sub-algorithms. In addition, the current private key $((DID_{j-1,1}, DID_{j-1,2}), (SID_{j-1,1}, SID_{j-1,2}))$ is also updated to $((DID_{j,1}, DID_{j,2}), (SID_{j,1}, SID_{j,2}))$.

- *Decrypt-1*: Given $DID_{j-1,1}$ and $SID_{j-1,1}$, this algorithm outputs $DID_{j,1}$, $SID_{j,1}$ and the temporary information TI_D.
- *Decrypt-2*: Given C, CT, TI_D, $DID_{j-1,2}$ and $SID_{j-1,2}$, this algorithm generates $DID_{j,2}$ and $SID_{j,2}$ while obtaining the encryption key K. Finally, the receiver can obtain the plain message msg by $D_K(CT)$ using the decryption function $D()$ of a symmetric cryptosystem.

3.2. *Security Notions of LR-CL-KE Scheme*

By the framework of LR-CL-KE scheme under the continual leakage model described in Section 3.1, an adversary $\mathcal{A}$ can obtain leakage information in four sub-algorithms: *Extract-1*, *Extract-2*, *Decrypt-1* and *Decrypt-2*. To represent the leakage information obtained by the adversary in the i-th *Initial key extract* round, two leakage functions $f_{IE,i}$ and $h_{IE,i}$ are chosen to model the adversary's abilities in *Extract-1* and *Extract-2*, respectively. Meanwhile, two leakage functions $f_{D,j}$ and $h_{D,j}$ are, respectively, chosen to model the adversary's ability in *Decrypt-1* and *Decrypt-2* of a user's j-th *Decrypt* round. It is worth mentioning that four leakage functions $f_{IE,i}$, $h_{IE,i}$, $f_{D,j}$ and $h_{D,j}$ can be efficiently computed while the output length of each leakage function is bounded by λ, where λ is the leakage parameter. That is, $|f_{IE,i}|$, $|h_{IE,i}|$, $|f_{D,j}|$, $|h_{D,j}| \leqslant \lambda$, where $|f|$ denotes the output length of leakage function f. We define the outputs of four leakage functions as follows.

- $\Lambda f_{IE,i} = f_{IE,i}(SK_{i-1,1}, params)$.
- $\Lambda h_{IE,i} = h_{IE,i}(SK_{i-1,2}, TI_{IE}, params)$.
- $\Lambda f_{D,j} = f_{D,j}(DID_{j-1,1}, SID_{j-1,1}, params)$.
- $\Lambda h_{D,j} = h_{D,j}(DID_{j-1,2}, SID_{j-1,2}, TI_D, params, skeys)$.

Here, *params* denotes the random values involved in the computation of four sub-algorithms *Extract-1*, *Extract-2*, *Decrypt-1* and *Decrypt-2*. Moreover, *skeys* denotes the symmetric encryption key. Note that TI_{IE} and TI_D are the outputs of *Extract-1* and *Decrypt-1*, respectively.

The adversary model of LR-CL-KE schemes consists of two types of adversaries, namely, Type I adversary (outsider) and Type II adversary (honest-but-curious KGC). Two types of adversaries are extended from the adversary model of CL-PKE schemes defined in Hwang *et al.* (2008), Tsai *et al.* (2015), Tsai and Tseng (2015) by adding the initial key extract leak query and decrypt leak query. This new model of LR-CL-KE schemes allows adversaries to learn partial information of the system secret key in the initial key extract phase and leak partial information of the user's private key in the decrypt phase. We describe two types of adversaries as follows.

- Type I Adversary (Outsider): This kind of adversary simulates the role of an outsider who can replace the public key of any user with another one chosen by herself/himself. That is, a Type I adversary may decide the secret key of any user with her/his choice. In addition, a Type I adversary may obtain the leakage information of a user's initial key in the decryption phase while leaking partial information of the KGC's system secret key in the *Initial key extract* phase.

- Type II Adversary (Honest-but-curious KGC): This kind of adversary simulates the role of the honest-but-curious KGC who owns the system's secret key and is disallowed to perform the public key replacement. In other words, a Type II adversary holds the initial key of any entity. In addition, a Type II adversary may obtain the leakage information of a user's secret key in the decryption phase.

In the following, a security game is used to model the security notions of the LR-CL-KE scheme under the continual leakage model. The security game below is played by the challenger $\mathcal{B}$ and an adversary $\mathcal{A}$.

Definition 2. (LR-CL-IND-CCA). We say that a LR-CL-KE scheme is semantically secure against indistinguishability under chosen ciphertext attack (LR-CL-IND-CCA) if no probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ (including Types I and II adversaries) has a non-negligible advantage in the following LR-CL-IND-CCA game played with a challenger $\mathcal{B}$.

- *Setup*. The challenger $\mathcal{B}$ takes a security parameter l as input, and runs the *Setup* algorithm to generate the first system secret key $(SK_{0,1}, SK_{0,1})$ and the public parameters *PP*. *PP* is sent to the adversary $\mathcal{A}$. In addition, if $\mathcal{A}$ is of Type II adversary, $\mathcal{B}$ also sends $(SK_{0,1}, SK_{0,1})$ to $\mathcal{A}$. If $\mathcal{A}$ is of Type I adversary, the first system secret key $(SK_{0,1}, SK_{0,1})$ is kept secret by $\mathcal{B}$.
- *Phase* 1. In this phase, the adversary $\mathcal{A}$ may adaptively issue the following queries:
  - *Initial key extract query* (*ID*). For the i-th *Initial key extract* query along with a user's identity *ID*, the challenger $\mathcal{B}$ uses the current system secret key $(SK_{i-1,1}, SK_{i-1,2})$ to generate the user's first initial key (DID_0, *QID*) while updating $(SK_{i-1,1}, SK_{i-1,2})$ to $(SK_{i,1}, SK_{i,2})$. Finally, $\mathcal{B}$ returns (DID_0, *QID*) to $\mathcal{A}$.
  - *Initial key extract leak query* $(f_{IE,i}, h_{IE,i}, i)$: By providing two leakage functions $f_{IE,i}$ and $h_{IE,i}$, $\mathcal{A}$ can issue the *Initial key extract leak query* only once for the i-th *Initial key extract query*. The challenger $\mathcal{B}$ computes the leakage information $(\Lambda f_{IE,i}, \Lambda h_{IE,i})$ and returns it to $\mathcal{A}$.
  - *Public key retrieve query* (*ID*). Upon receiving this query along with an identity *ID*, $\mathcal{B}$ returns the corresponding public key $PID = (QID, RID)$.
  - *Public key replace query* (*ID*, $PID' = (QID', RID')$). Upon receiving this query along with (*ID*, PID'), $\mathcal{B}$ records the replacement. It means that the adversary $\mathcal{A}$ has replaced the user's public key with $PID' = (QID', RID')$.
  - *Secret key extract query* (*ID*). When the challenger $\mathcal{B}$ receives this query along with an identity *ID*, $\mathcal{B}$ returns the secret key SID_0. Moreover, the query is forbidden if *Public key replace query* (*ID*) has been previously queried in this game.
  - *Decrypt query* (*ID*, C). For the j-th *Decrypt* round, upon receiving this query along with an identity *ID* and a ciphertext C, the challenger $\mathcal{B}$ uses the user's current private key $(DID_{j-1} = (DID_{j-1,1}, DID_{j-1,2})$, $SID_j = (SID_{j-1,1}, SID_{j-1,2}))$ to generate the encryption key K by running two sub-algorithms *Decrypt-1* and *Decrypt-2*. The challenger $\mathcal{B}$ then returns K to $\mathcal{A}$. It is worth mentioning that the current private key $(DID_{j-1} = (DID_{j-1,1}, DID_{j-1,2})$,

$SID_j = (SID_{j-1,1}, SID_{j-1,2})$) is also updated to ($DID_j = (DID_{j,1}, DID_{j,2})$, $SID_j = (SID_{j,1}, SID_{j,2})$).

- *Decrypt leak query* $(f_{D,j}, h_{D,j}, j)$: By providing two leakage functions $f_{D,j}$ and $h_{D,j}$, $\mathcal{A}$ can issue the *Decrypt leak query* only once for the j-th *Decrypt query*. $\mathcal{B}$ computes the leakage information $(\Lambda f_{D,j}, \Lambda h_{D,j})$ and returns it to $\mathcal{A}$. In addition, if $\mathcal{A}$ is of Type II adversary (honest-but-curious KGC), $(\Lambda f_{D,j}, \Lambda h_{D,j})$ includes only the leakage information of a user's secret key $(SID_{j-1,1}, SID_{j-1,2})$ since $\mathcal{A}$ knows the initial key of any entity. If $\mathcal{A}$ is of Type I adversary (outsider), the adversary may obtain the leakage information of a user's initial key $(DID_{j-1,1}, DID_{j-1,2})$ since an outsider owns the secret key of any entity.

– *Challenge*. The adversary $\mathcal{A}$ chooses a target identity ID^* and a plaintext pair (msg_0^*, msg_1^*) to the challenger $\mathcal{B}$. Two restrictions are described as follows.

1. If $\mathcal{A}$ is of Type I adversary (outsider), the *Initial key extract query* (ID^*) is not queried in *Phase* 1.
2. If $\mathcal{A}$ is of Type II adversary (honest-but-curious KGC), it is disallowed to issue the queries on the *Secret value extract query* and *Public key replace query* on ID^* in *Phase* 1.

The challenger $\mathcal{B}$ chooses a random $\beta \in \{0, 1\}$ and computes $C^* = E(PP, ID^*, msg_\beta^*, PK_{ID^*})$ by running the *Encrypt* algorithm. Then $\mathcal{B}$ sends C^* to $\mathcal{A}$. Here PK_{ID^*} is the public key of the identity ID^*.

– *Guess*. The adversary $\mathcal{A}$ outputs $\beta' \in \{0, 1\}$ and wins this game if $\beta' = \beta$.

In the LR-CL-IND-CCA game above, we call the adversary $\mathcal{A}$ as a LR-CL-IND-CCA adversary. We define the adversary $\mathcal{A}$'s advantage in attacking the LR-CL-KE scheme as $Adv_A(l) = |Pr[\beta = \beta'] - \frac{1}{2}|$.

Remark. The LR-CL-IND-CCA game defined above models the security notion of LR-CL-KE scheme against non-adaptive chosen ciphertext attacks (CCA1). For the security notion against adaptive chosen ciphertext attack (CCA2), a new *Phase 2* is inserted between the *Challenge phase* and *Guess phase*. In the *Phase 2*, $\mathcal{A}$ may issue further queries as in *Phase* 1 while a restriction is that $\mathcal{A}$ cannot make a *Decrypt query* on the challenge ciphertext $C^* = E(Parms, ID^*, msg_\beta^*, PK_{ID^*})$. It is worth mentioning that our LR-CL-KE scheme is secure against CCA1 under the continual leakage model, but it can't achieve CCA2 security. The reason will be discussed in Section 5.

4. The Proposed LR-CL-KE Scheme

In the following, we propose the first LR-CL-KE scheme, denoted by Π. As the framework defined in Section 3.1, the LR-CL-KE scheme consists of seven algorithms as follows:

– *Setup*: Given a security parameter l, the KGC first generates two multiplicative groups G and G_T of prime order p and then randomly picks a generator g of G. Let $e : G \times G \to G_T$ be an admissible bilinear pairing. The KGC also selects a symmetric cryptosystem with encryption function $E()$ and decryption function $D()$. The

KGC runs the following steps to generate the first system secret key $(SK_{0,1}, SK_{0,1})$ and the public parameters *PP*:

(1) Randomly pick $x \in Z_p^*$ and compute $X = g^x$ and $X_T = e(g^x, g)$.

(2) Randomly pick $\alpha \in Z_p^*$ and set the first system secret key $(SK_{0,1}, SK_{0,1}) = (g^\alpha, X \cdot g^\alpha)$.

(3) Randomly pick $u_{i0}, u_{i1} \in Z_q^*$ and compute $U_0 = g^{u_{i0}}$ and $U_1 = g^{u_{i1}}$.

(4) Publish $PP = (G, G_T, e, p, g, X_T, U_0, U_1, E, D)$.

– *Initial key extract*: For the i-th *Initial key extract* round, upon receiving a user's identity *ID*, the KGC generates the first initial key (DID_0, QID) of the user by running two sub-algorithms *Extract-1* and *Extract-2* as follows. In addition, the current system secret key $(SK_{i-1,1}, SK_{i-1,2})$ is also updated to $(SK_{i,1}, SK_{i,2})$.

- *Extract-1*: The KGC uses $(ID, SK_{i-1,1})$ to compute the temporary information TI_E and *QID* as follows.
  (1) Choose two random numbers $\gamma, a \in Z_p^*$.
  (2) Compute $QID = g^\gamma$ and $SK_{i,1} = SK_{i-1,1} \cdot g^a$.
  (3) Compute the temporary information $TI_E = SK_{i,1} \cdot (U_0 \cdot U_1^{ID})^\gamma$.
- *Extract-2*: The KGC uses $(TI_E, SK_{i-1,2})$ to generate DID_0 as follows.
  (1) Compute $SK_{i,2} = SK_{i-1,2} \cdot g^{-a}$.
  (2) Compute $DID_0 = SK_{i,2} \cdot TI_E$.

Finally, the KGC sends the first initial key $(DID_0, QID) = (X \cdot (U_0 \cdot U_1^{ID})^\gamma, g^\gamma)$ to the user via a secure channel. Note that the user may validate the correctness of (DID_0, QID) by checking the equality $e(g, DID_0) = X_T \cdot e(QID, U_0 \cdot U_1^{ID})$.

– *Set secret value*: A user with identity *ID* chooses a random number $z \in Z_p^*$ and computes the first secret key SID_0 and the partial public key *RID*, where $(SID_0, RID) = (g^z, e(g^z, g))$.

– *Set private key*: Given the initial key (DID_0, QID) and the secret key SID_0, the user sets her/his private key by the steps below:

- Select two random numbers $\beta, \omega \in Z_p^*$.
- Compute the first private key $((DID_{0,1}, DID_{0,2}) = (g^\beta, DID_0 \cdot g^{-\beta}), (SID_{0,1}, SID_{0,2}) = (g^\omega, SID_0 \cdot g^{-\omega}))$.

– *Set public key*: Given the initial key (DID_0, QID) and the partial public key *RID*, the user with identity *ID* sets her/his public key $PID = (QID, RID) = (g^\gamma, e(g^z, g))$.

– *Encrypt*: Given the public key $PID = (QID, RID)$ of a receiver with identity *ID*, the sender runs the following steps to encrypt the plaintext msg:

(1) Randomly choose $k \in Z_p^*$.

(2) Compute $C = g^k$, $K_1 = (RID)^k = e(g^z, g)k$ and $K_2 = (X_T \cdot e(QID, U_0 \cdot U_1^{ID}))^k$.

(3) Set the encryption key $K = K_1 \oplus K_2$.

(4) Generate $CT = E_K(msg)$.

Finally, the sender returns the ciphertext (C, CT) to the receiver.

– *Decrypt*: For the j-th *Decrypt* round, given the ciphertext (C, CT), the receiver with identity *ID* uses the current private key $((DID_{j-1,1}, DID_{j-1,2}), (SID_{j-1,1}, SID_{j-1,2}))$ to recover the plaintext msg by performing two sub-algorithms as follows. In addition, the current private key $((DID_{j-1,1}, DID_{j-1,2}), (SID_{j-1,1}, SID_{j-1,2}))$ is also updated to $((DID_{j,1}, DID_{j,2}), (SID_{j,1}, SID_{j,2}))$.

- *Decrypt-1*: The receiver uses $DID_{j-1,1}$ and $SID_{j-1,1}$ to compute the temporary information TI_1 and TI_2 by the following steps:
  (1) Choose two random numbers $b, c \in Z_p^*$.
  (2) Update $DID_{j,1} = DID_{j-1,1} \cdot g^b$ and $SID_{j,1} = SID_{j-1,1} \cdot g^c$.
  (3) Compute $TI_1 = e(C, SID_{j,1})$ and $TI_2 = e(C, DID_{j,1})$.
- *Decrypt-2*: Given C, CT, TI_1 and TI_2, the receiver uses $DID_{j-1,2}$ and $SID_{j-1,2}$ to return plaintext msg by the following steps:
  (1) Compute $DID_{j,2} = DID_{j-1,2} \cdot g^{-b}$ and $SID_{j,2} = SID_{j-1,2} \cdot g^{-c}$.
  (2) Compute $K_1' = TI_1 \cdot e(C, SID_{j,2})$ and $K_2' = TI_2 \cdot e(C, DID_{j,2})$.
  (3) The encryption key is computed by $K' = K_1' \oplus K_2'$.
  (4) Obtain the plaintext $msg = D_{K'}(CT)$ by using a symmetric cryptosystem.

In the following, we show the correctness of recovering the encryption key.

$$\begin{aligned}
K &= K_1 \oplus K_2 \\
&= (RID)^k \oplus \big(X_T \cdot e(QID, U_0 \cdot U_1^{ID})\big)k \\
&= e(g^z, g)^k \oplus (X_T \cdot e\big(g^\gamma, U_0 \cdot U_1^{ID})\big)^k \\
&= e\big(g^k, g^z\big) \oplus \big(e\big(g^x, g\big) \cdot e\big(g^\gamma, U_0 \cdot U_1^{ID}\big)\big)^k \\
&= e\big(g^k, SID_0\big) \oplus (e\big(g, g^x\big) \cdot e\big(g, (U_0 \cdot U_1^{ID})^\gamma\big)^k \\
&= e\big(g^k, SID_0\big) \oplus (e\big(g, g^x \cdot (U_0 \cdot U_1^{ID})^\gamma\big)^k \\
&= e\big(g^k, SID_0\big) \oplus e\big(g^k, X \cdot (U_0 \cdot U_1^{ID})^\gamma\big) \\
&= e(C, SID_0) \oplus e(C, DID_0) \\
&= e\big(C, SID_0 \cdot g^{-\omega} \cdot g^{\omega}\big) \oplus e\big(C, DID_0 \cdot g^{-\beta} \cdot g^{\beta}\big) \\
&= e\big(C, g^{\omega}\big) \cdot e\big(C, SID_0 \cdot g^{-\omega}\big) \oplus e\big(C, g^{\beta}\big) \cdot e\big(C, DID_0 \cdot g^{-\beta}\big) \\
&= e(C, SID_{0,1}) \cdot e(C, SID_{0,2}) \oplus e(C, DID_{0,1}) \cdot e(C, DID_{0,2}) \\
&= TI_1 \cdot e(C, SID_{0,2}) \oplus TI_2 \cdot e(C, DID_{0,2}) \\
&= K_1' \oplus K_2' \\
&= K'.
\end{aligned}$$

5. Security Analysis

As the aforementioned LR-CL-IND-CCA game in Definition 2, there are two types of adversaries, Type I (outsider) and Type II (honest-but-curious KGC). In the section, we present the security analysis of the proposed LR-CL-KE scheme under the continual leakage model for both Type I and Type II adversaries. Indeed, our proposed LR-CL-KE scheme achieves only the CCA1 security (See *Remark in Section 3), but it can't achieve the CCA2 security. The reason is that an adversary, given the challenge ciphertext C^* with encryption key $K = K_1 \oplus K_2$, can obtain the entire K via the leakage information. That is, the adversary may ask the *Decrypt query* with input $(C^*)^2 \neq C^*$ repeatedly to collect the leakage information about K_1^2 and K_2^2 by *Decrypt leak query*. Then the adversary can reconstruct K_1 and K_2 from K_1^2 and K_2^2, respectively. Finally, an adversary may compute the encryption key $K = K_1 \oplus K_2$. To our best knowledge, no LR-CL-KE scheme under continual leakage model can achieve the CCA2 security.

In the following, we first introduce the non-leakage version of our LR-CL-KE scheme, denoted by Π_{NL}. Then we prove that the non-leakage version Π_{NL} is CL-IND-CCA secure in the generic bilinear group model. Next, based on the security of the non-leakage version Π_{NL}, we prove that the proposed LR-CL-KE scheme is LR-CL-IND-CCA secure under the continual leakage model.

The non-leakage version Π_{NL} of our LR-CL-KE scheme consists of seven algorithms as follows:

– *Setup$_{NL}$*: In this algorithm, the KGC generates the system secret key $X = g^x$, where x is a random number picked from Z_p^*. Moreover, the public parameters $PP = (G, G_T, e, p, g, X_T, U_0, U_1)$ are identical to those in the proposed LR-CL-KE scheme. Finally, the KGC publishes the public parameters PP.
– *Initial key extract$_{NL}$*: The KGC generates the initial key $(DID, QID) = (X \cdot (U_0 \cdot U_1^{ID})^\gamma, g^\gamma)$ of a user with identity *ID*, where γ is picked from Z_p^* randomly. The KGC then sends the initial key (DID, QID) to the user via a secure channel.
– *Set secret value$_{NL}$*: A user chooses a random number z in Z_p^*, and computes the user's secret key $SID = g^z$ and the associated partial public key $RID = e(g^z, g)$.
– *Set private key$_{NL}$*: A user sets her/his private key $(DID, SID) = (X \cdot (U_0 \cdot U_1^{ID})^\gamma, g^z)$.
– *Set public key$_{NL}$*: A user sets her/his public key $PID = (QID, RID)$.
– *Encrypt$_{NL}$*: Given the public key $PID = (QID, RID)$ of a user with identity *ID*, the sender randomly chooses $k \in Z_p^*$, and then computes $C = g^k$, $K_1 = (RID)^k$ and $K_2 = (X_T \cdot e(QID, U_0 \cdot U_1^{ID}))^k$. The encryption key is $K = K_1 \oplus K_2$. The ciphertext $(C, CT = E_K(msg))$ is sent to the receiver, where msg is the plaintext.
– *Decrypt$_{NL}$*: Upon receiving a user's identity *ID* and the ciphertext (C, CT), the receiver uses the private key (DID, SID) to get the encryption key $K = K_1 \oplus K_2$ by computing $K_1 = e(C, SID)$ and $K_2 = e(C, DID)$. Then she/he can decrypt the plaintext $msg = D_K(CT)$.

In Theorems 1 and 2, we prove that the non-leakage version Π_{NL} of our LR-CL-KE scheme is CL-IND-CCA secure against Types I and II adversaries, respectively. Moreover, in Theorems 3 and 4, we prove that our LR-CL-KE scheme is LR-CL-IND-CCA secure against Types I and II adversaries, respectively.

Theorem 1. *In the generic bilinear group model, the non-leakage version Π_{NL} of our LR-CL-KE scheme is CL-IND-CCA secure against Type I adversary (outsider).*

Proof. Let $\mathcal{A}_{NL-I}$ be Type I adversary (outsider) who can break the non-leakage version Π_{NL}. The adversary $\mathcal{A}_{NL-I}$ can adaptively issue all the queries at most q times in total.

The advantage that $\mathcal{A}_{NL-I}$ breaks Π_{NL} is bounded by the success probability of $\mathcal{A}_{NL-I}$ in the game g_{NL-I} which is played by a challenger $\mathcal{B}$ and the adversary $\mathcal{A}_{NL-I}$ as follows:
***Game** g_{NL-I}: In the game g_{NL-I}, there are four phases, namely, *Setup*, *Phase* 1, *Challenge* and *Guess*, which are described as follows.

- *Setup*: The challenger $\mathcal{B}$ constructs two lists L_G and L_T to record the elements in the groups G and G_T, respectively.
  - The list L_G consists of elements of the form $(P_{G,m,n,r}, \phi_{G,m,n,r})$. Each $P_{G,m,n,r}$ is a multivariate polynomial consists of a finite numbers of variates in G with coefficients in Z_p. For a multivariate polynomial $P_{G,m,n,r}$, the challenger $\mathcal{B}$ uses a bit string $\phi_{G,m,n,r}$ to communicate with $\mathcal{A}_{NL-I}$. The first index "G" in $P_{G,m,n,r}$ or $\phi_{G,m,n,r}$ indicates that $P_{G,m,n,r}$ or $\phi_{G,m,n,r}$ is an element in G and is an element in G_T if the index G is replaced by T. Moreover, the second index "m" indicates the type of query. The third and fourth indices "n" and "r" indicate the r-th element in G/G_T which appeared in the n-th query. Four tuples $(g, \phi_{G,I,1,1})$, $(X, \phi_{G,I,1,2})$, $(U_0, \phi_{G,I,1,3})$ and $(U_1, \phi_{G,I,1,4})$ are initially added in the list L_G.
  - The list L_T is used to record the elements with the form of $(P_{T,m,n,r}, \phi_{T,m,n,r})$. The meanings of all indexes of $P_{T,m,n,r}$ are the same with the descriptions of $P_{G,m,n,r}$ earlier. $P_{T,m,n,r}$ is a multivariate polynomial with coefficients in Z_p and variates in G or G_T. For each multivariate polynomial $P_{T,m,n,r}$, the challenger $\mathcal{B}$ uses the bit string $\phi_{T,m,n,r}$ to communicate with $\mathcal{A}_{NL-I}$. The tuple $(X_T, \phi_{T,I,1,1})$ is initially added into L_T.

  Moreover, two additional lists L_{IK} and L_{SK} are constructed to record the user's initial key and the user's secret key, respectively. More precisely, the elements in L_{IK} are of the form (*ID*, *DID*, *QID*) and the elements in L_{SK} are of the form (*ID*, *SID*, *RID*), where *ID* is in Z_p, and *DID*, *QID*, *SID* and *RID* are multivariate polynomials. At the end of this phase, the challenger $\mathcal{B}$ sends the public parameters *PP* to $\mathcal{A}_{NL-I}$ (using the form of bit strings).
- *Phase* 1: In this phase, $\mathcal{A}_{NL-I}$ can adaptively issue the following queries at most q times totally.
  - *Group G query* Q_G ($\phi_{G,Q,i,1}$, $\phi_{G,Q,i,2}$, *operation*): For the i-th group query Q_G, $\mathcal{A}_{NL-I}$ queries $\mathcal{B}$ along with two bit strings $(\phi_{G,Q,i,1}, \phi_{G,Q,i,2})$ and an *operation* (multiplication or division). The challenger $\mathcal{B}$ performs the following steps.
    - (i) $\mathcal{B}$ first translates two bit strings $\phi_{G,Q,i,1}$ and $\phi_{G,Q,i,2}$ into two polynomials $P_{G,Q,i,1}$ and $P_{G,Q,i,2}$, respectively, in the following way. $\mathcal{B}$ tries to find a pair $(P_{G,m,n,r}, \phi_{G,m,n,r})$ in L_G such that $\phi_{G,m,n,r} = \phi_{G,Q,i,1}$. If so, $\mathcal{B}$ sets $P_{G,Q,i,1} = P_{G,m,n,r}$. Otherwise, $\mathcal{B}$ defines a new variate $S_{G,Q,i,1}$ in G, sets $P_{G,Q,i,1} = S_{G,Q,i,1}$, and records $(P_{G,Q,i,1}, \phi_{G,Q,i,1})$ in L_G. Similarly, $\mathcal{B}$ also translates the bit string $\phi_{G,Q,i,2}$ to $P_{G,Q,i,2}$.
    - (ii) $\mathcal{B}$ computes the polynomial $P_{G,Q,i,3} = P_{G,Q,i,1} + P_{G,Q,i,2}$ if the *operation* is multiplication, or $P_{G,Q,i,3} = P_{G,Q,i,1} - P_{G,Q,i,2}$ if the *operation* is division.
    - (iii) $\mathcal{B}$ uses $P_{G,Q,i,3}$ to find an element $(P_{G,m,n,r}, \phi_{G,m,n,r})$ in L_G such that $P_{G,m,n,r} = P_{G,Q,i,3}$. If so, $\mathcal{B}$ returns $\phi_{G,m,n,r}$ to $\mathcal{A}_{NL-I}$. Otherwise, $\mathcal{B}$ randomly chooses a bit string, denoted by $\phi_{G,Q,i,3}$, which is distinct from all bit

strings recorded in L_G and L_T. Finally, $\mathcal{B}$ records $(P_{G,Q,i,3}, \phi_{G,Q,i,3})$ in L_G and returns $\phi_{G,Q,i,3}$ to $\mathcal{A}_{NL-I}$.

Note that the polynomials $P_{G,Q,i,1}$, $P_{G,Q,i,2}$ and $P_{G,Q,i,3}$ mentioned above are recorded in the list L_G.

- *Group G_T query $Q_T(\phi_{T,Q,i,1}, \phi_{T,Q,i,2}, operation)$*: For the i-th group query Q_T, $\mathcal{A}_{NL-I}$ queries $\mathcal{B}$ along with two bit strings $(\phi_{T,Q,i,1}, \phi_{T,Q,i,2})$ and an *operation* (multiplication or division). The process of this query is similar to that of the Group G query Q_G. $\mathcal{B}$ finally returns $\phi_{T,Q,i,3}$ to $\mathcal{A}_{NL-I}$. After this query, the polynomials $P_{T,Q,i,1}$, $P_{T,Q,i,2}$ and $P_{T,Q,i,3}$ are recorded in L_T.
- *Pairing query $Q_P(\phi_{G,P,i,1}, \phi_{G,P,i,2})$*: For the i-th pairing query Q_P, $\mathcal{A}_{NL-I}$ takes as input two bit strings $\phi_{G,P,i,1}$ and $\phi_{G,P,i,2}$. $\mathcal{B}$ performs the following steps:
  (i) $\mathcal{B}$ first translates two bit strings $\phi_{G,P,i,1}$ and $\phi_{G,P,i,2}$ to two polynomials $P_{G,P,i,1}$ and $P_{G,P,i,2}$, respectively. This step is similar to the Step 1 of the *Group G query Q_G*.
  (ii) $\mathcal{B}$ computes the polynomial $P_{T,P,i,1} = P_{G,P,i,1} \cdot P_{G,P,i,2}$.
  (iii) $\mathcal{B}$ uses $P_{T,P,i,1}$ to find $(P_{T,m,n,r}, \phi_{T,m,n,r})$ in L_T such that $P_{T,m,n,r} = P_{T,P,i,1}$. If so, $\mathcal{B}$ returns $\phi_{T,m,n,r}$ to $\mathcal{A}_{NL-I}$. Otherwise, $\mathcal{B}$ randomly chooses a bit string, denoted by $\phi_{T,P,i,1}$, which is distinct from all bit strings recorded in L_G and L_T. Then $\mathcal{B}$ records $(P_{T,P,i,1}, \phi_{T,P,i,1})$ in L_T and returns $\phi_{T,P,i,1}$ to $\mathcal{A}_{NL-I}$.

  Note that the polynomials $P_{G,P,i,1}$, $P_{G,P,i,2}$ and $P_{T,P,i,1}$ are recorded in the list L_G or L_T after this query.
- *Initial key extract query $Q_{IE}(ID_{IE,i})$*: For the i-th *Initial key extract query*, $\mathcal{A}_{NL-I}$ queries $\mathcal{B}$ along with a user's identity $ID_{IE,i} \in Z_p^*$. $\mathcal{B}$ tries to find $ID_{IE,i}$ in L_{IK}. If so, $\mathcal{B}$ obtains the corresponding multivariate polynomials $P_{G,IE,i,1}$ and $P_{G,IE,i,2}$ of the user's initial key *DID* and *QID*. $\mathcal{B}$ then returns two corresponding bit strings $\phi_{G,IE,i,1}$ and $\phi_{G,IE,i,2}$ to $\mathcal{A}_{NL-I}$. Otherwise, $\mathcal{B}$ performs three steps as follows:
  (i) $\mathcal{B}$ selects one variate $T_{G,IE,i,2}$ in G (which denotes the *QID* of $ID_{IE,i}$) and sets $P_{G,IE,i,2}$=$T_{G,IE,i,2}$. Moreover, $\mathcal{B}$ randomly chooses a bit string, denoted by $\phi_{G,IE,i,2}$, which is distinct from all bit strings recorded in L_G and L_T. Then $\mathcal{B}$ records $(P_{G,IE,i,2}, \phi_{G,IE,i,2})$ in L_G.
  (ii) $\mathcal{B}$ computes the polynomial $P_{G,IE,i,1} = X + (U_0 + ID_{IE,i} \cdot U_1) \cdot T_{G,IE,i,2}$, which represents the *DID* of $ID_{IE,i}$.
  (iii) Finally, $\mathcal{B}$ chooses a bit string $\phi_{G,IE,i,1}$, which is distinct from all bit strings recorded in L_G and L_T. Then $\mathcal{B}$ records $(P_{G,IE,i,1}, \phi_{G,IE,i,1})$ in L_G and returns $(\phi_{G,IE,i,1}, \phi_{G,IE,i,2}\ )$ to $\mathcal{A}_{NL-I}$.

  The challenger $\mathcal{B}$ also records a tuple $(ID_{IE,i}, P_{G,IE,i,1}, P_{G,IE,i,2})$ in the list L_{IK}.
- *Secret key extract query $Q_{SE}(ID_{SE,i})$*: For the i-th *Secret key extract query*, $\mathcal{A}_{NL-I}$ queries the challenger $\mathcal{B}$ along with an identity $ID_{SE,i}$. $\mathcal{B}$ performs the following steps and finally outputs the bit strings $(\phi_{T,SE,i,1}, \phi_{T,SE,i,2})$, which represent the secret key (*SID*, *RID*) to $\mathcal{A}_{NL-I}$:
  (i) The challenger $\mathcal{B}$ checks whether the secret key of identity $ID_{SE,i}$ has been recorded in L_{SK}. If so, $\mathcal{B}$ returns the bit strings $(\phi_{T,SE,i,1}, \phi_{T,SE,i,2})$, which

represents the secret key (SID, RID) to $\mathcal{A}_{NL-I}$. Otherwise, $\mathcal{B}$ defines a new variate $T_{G,SE,i,2}$ in G and sets $P_{G,SE,i,1} = T_{G,SE,i,1}$, which represents the SID of $ID_{SE,i}$. Moreover, $\mathcal{B}$ randomly chooses a new bit string, denoted by $\phi_{G,SE,i,1}$, which is distinct from all bit strings recorded in L_G and L_T. Then $\mathcal{B}$ records $(P_{G,SE,i,1}, \phi_{G,SE,i,1})$ in L_G.

(ii) $\mathcal{B}$ sets the polynomial $P_{T,SE,i,2} = T_{G,SE,i,1} \cdot g$, which represents the RID for $ID_{SE,i}$.

(iii) Finally, $\mathcal{B}$ first randomly chooses a new bit string, denoted by $\phi_{T,SE,i,2}$, which is distinct from all bit strings recorded in L_G and L_T. Then $\mathcal{B}$ records $(P_{T,SE,i,2}, \phi_{T,SE,i,2})$ in L_T and returns $(\phi_{G,SE,i,1}, \phi_{T,SE,i,2})$ to $\mathcal{A}_{NL-I}$.

The challenger $\mathcal{B}$ also records $(ID_{SE,i}, P_{G,SE,i,1}, P_{T,SE,i,2})$ in the list L_{SK}.

- *Public key retrieve query* $Q_{PK}(ID_{PK,i})$: Upon receiving the i-th *Public key retrieve* query with an identity $ID_{PK,i} \in Z_p^*$ as input, the challenger $\mathcal{B}$ performs the following three steps:
  (i) $\mathcal{B}$ checks whether $ID_{PK,i}$ has been recorded in L_{IK}. If so, $\mathcal{B}$ obtains the corresponding polynomial of QID for $ID_{PK,i}$ in L_{IK}. Otherwise, $\mathcal{B}$ performs the *Initial key extract query* along with identity $ID_{PK,i}$ to generate the polynomial of QID.
  (ii) $\mathcal{B}$ also checks whether $ID_{PK,i}$ has been recorded in L_{SK}. If so, $\mathcal{B}$ obtains the corresponding polynomial of RID for $ID_{PK,i}$ in L_{SK}. Otherwise, $\mathcal{B}$ performs the *Secret key extract query* with identity $ID_{PK,i}$ to generate the polynomial of RID for $ID_{PK,i}$.
  (iii) Finally, C returns two bit strings of polynomials representing QID and RID by searching L_G and L_T, respectively.
- *Public key replace query* $Q_{PR}(ID_{PR,i}, \phi_{T,PR,i,2})$: By this query, the adversary $\mathcal{A}_{NL-I}$ is allowed to use the bit string $\phi_{T,PR,i,2}$ to replace the partial public key RID of a user with identity $ID_{PR,i}$. That is, $\mathcal{A}_{NL-I}$ can select a valid secret key SID (i.e. bit string $\phi_{T,PR,i,2}$) by herself/himself and set the corresponding RID. $\mathcal{B}$ then records this replacement. More precisely, upon receiving this query, $\mathcal{B}$ first translates $\phi_{T,PR,i,2}$ into the corresponding polynomial $P_{T,PR,i,2}$ by the list L_T. Since $\mathcal{A}_{NL-I}$ has the ability to generate the user's secret key by asking the group oracles, $\mathcal{B}$ can obtain the polynomial $P_{G,PR,i,1}$ by searching $P_{T,PR,i,2} = P_{G,PR,i,1} \cdot g$ in the list L_G. The challenger $\mathcal{B}$ then updates the user's secret key $(ID_{PR,i}, SID_{PR,i}, RID_{PR,i}) = (ID_{PR,i}, P_{G,PR,i,1}, P_{T,PR,i,2})$ in L_{SK}.
- *Decrypt query* $Q_D(ID_{D,i}, C_i, CT_i)$: For the i-th Decrypt round, when $\mathcal{A}_{NL-I}$ queries $\mathcal{B}$ along with a user's identity $ID_{D,i}$ and a ciphertext pair (C_i, CT_i), $\mathcal{B}$ performs the following two parts to obtain the encryption key K:
  (1) When $\mathcal{B}$ receives the query, $\mathcal{B}$ first obtains the user's initial key DID and secret key SID from the lists L_{IK} and L_{SK}, respectively, by the following procedures:
    (i) $\mathcal{B}$ uses $ID_{D,i}$ to find the user's initial key DID in the list L_{IK}. If so, $\mathcal{B}$ obtains DID in L_{IK}. Otherwise, $\mathcal{B}$ issues the query $Q_{IE}(ID_{D,i})$ to obtain DID.

(ii) $\mathcal{B}$ uses $ID_{D,i}$ to find the user's secret key *SID* in the list L_{SK}. If so, $\mathcal{B}$ obtains *SID* in L_{SK}. Otherwise, $\mathcal{B}$ issues the query $Q_{SE}(ID_{D,i})$ to obtain *SID*.

(iii) Hence, $\mathcal{B}$ has obtained the polynomials $P_{G,IE,k,1}$ and $P_{G,SE,l,1}$, which represent *DID* and *SID*, respectively.

(2) The challenger $\mathcal{B}$ obtains the encryption key K by performing the following steps:

(i) $\mathcal{B}$ checks whether the corresponding polynomial of the ciphertext C_i has been recorded in the list L_G. If so, $\mathcal{B}$ obtains the polynomial $P_{G,D,i,3}$. Otherwise, $\mathcal{B}$ defines a new variate $T_{G,D,i,3}$ in G and sets $P_{G,D,i,3} = T_{G,D,i,3}$. Moreover, $\mathcal{B}$ randomly chooses a bit string, denoted by $\phi_{G,D,i,3}$, which is distinct from all bit strings in L_G and L_T.

(ii) $\mathcal{B}$ computes the polynomial $P_{T,D,i,1} = P_{G,SE,l,1} \cdot T_{G,D,i,1}$ (which denotes K_1) and the polynomial $P_{T,D,i,2} = P_{G,IE,l,1} \cdot T_{G,D,i,1}$ (which denotes K_2). $\mathcal{B}$ uses $P_{T,D,i,1}$ and $P_{T,D,i,2}$ to respectively find ($P_{T,m,j,r}$, $\phi_{T,m,j,r}$) and ($P_{T,m,j,2}$, $\phi_{T,m,j,2}$) in L_T (i.e. $PT, m, j, r = P_{T,D,i,1}$ and $P_{T,m,j,2} = P_{T,D,i,2}$). If so, $\mathcal{B}$ then sets the $\phi_{T,D,i,1} = \phi_{T,m,j,r}$ and $\phi_{T,D,i,2} = \phi_{T,m,j,2}$. Otherwise, $\mathcal{B}$ randomly chooses two bit strings, denoted by $\phi_{T,D,i,1}$ and $\phi_{T,D,i,2}$, which represent the bit strings of K_1 and K_2, respectively. Then $\mathcal{B}$ records ($P_{T,D,i,1}$, $\phi_{T,D,i,1}$) and ($P_{T,D,i,2}$, $\phi_{T,D,i,2}$) in L_T.

Finally, $\mathcal{B}$ computes the bit string $\phi_{T,D,i,4} = \phi_{T,D,i,1} \oplus \phi_{T,D,i,2}$ (which denotes K). At the end of this query, the challenger $\mathcal{B}$ obtains the plaintext $msg_i = D_K(CT_i)$ by using the decryption key K. Finally, $\mathcal{B}$ returns msg_i to $\mathcal{A}_{NL-I}$.

– *Challenge*: The adversary $\mathcal{A}_{NL-I}$ gives a target identity ID^* and a plaintext pair (msg_0^*, msg_1^*) to $\mathcal{B}$. Because $\mathcal{A}_{NL-I}$ is an outsider, ID^* is disallowed to be queried in the *Initial key extract query* of *Phase* 1. The challenger $\mathcal{B}$ first chooses a random bit $\beta \in \{0, 1\}$, then $\mathcal{B}$ defines a new variate $T_{G,C,1,3}$ in G and sets $P_{G,C,1,3} = T_{G,C,1,3}$ (which denotes C^*). Moreover, $\mathcal{B}$ randomly chooses a bit string, denoted by $\phi_{G,C,i,3}$. Afterwards, $\mathcal{B}$ obtains K by the same steps described in the second part of the *Decrypt query*. At the end of this phase, the challenger $\mathcal{B}$ computes the ciphertext $CT^* = E_K(msg_\beta^*)$. Finally, $\mathcal{B}$ returns C^* and CT^* to $\mathcal{A}_{NL-I}$.

– *Guess*: The adversary $\mathcal{A}_{NL-I}$ outputs $\beta' \in \{0, 1\}$. If $\beta' = \beta$, we say that the adversary $\mathcal{A}_{NL-I}$ wins the game g_{NL-I}.

Here, both the adversary $\mathcal{A}_{NL-I}$ and the challenger $\mathcal{B}$ have completed the game g_{NL-I}. Before evaluating the probability of $\mathcal{A}_{NL-I}$ winning the game g_{NL-I}, we first define several notations and restrictions as follows.

(1) In the *Phase* 1, $\mathcal{A}_{NL-I}$ may issue eight kinds of queries Q_G, Q_T, Q_P, Q_{IE}, Q_{SE}, Q_{PK}, Q_{PR}, Q_D. We define several collections (sets) as follows:

- $\{S\}$: The collection of all used variates $S_{G,Q,i,j}$ in the query Q_G and $S_{G,P,i,j}$ in the query Q_P.
- $\{V\}$: The collection of all used variates $V_{T,Q,i,j}$ in the query Q_T.

- $\{T\}$: The collection of all used variates $T_{G,IE,i,2}$ in the query Q_{IE}, $T_{G,D,i,3}$ in the query Q_D and $T_{G,SE,i,1}$ in the query Q_{SE}.
- $\{PG\}$: The collection of all used polynomials $P_{G,Q,i,k}$, $P_{G,IE,i,k}$ and $P_{G,D,i,k}$ in the *Phase* 1.
- $\{PT\}$: The collection of all used polynomials $P_{T,Q,i,k}$ and $P_{T,P,i,k}$ in the *Phase* 1.

(2) Let q_O denote the total number of three queries Q_G, Q_T and Q_P while q_{IE}, q_{SE}, q_{PK}, q_{PR} and q_D, respectively, represent the numbers of Q_{IE}, Q_{SE}, Q_{PK}, Q_{PR} and Q_D. Note that $\mathcal{A}_{NL-I}$ can issue all kinds of queries at most q times in total. Hence, we have $q \geqslant q_O + q_{IE} + q_{SE} + q_{PK} + q_{PR} + q_D$. Let $|L_G|$ and $|L_T|$ be the total numbers of elements in L_G and L_T, respectively. Therefore, $|L_G|$+ $|L_T|$ is bounded by $6q$ due to the following reasons:

- For each query of Q_G, Q_T or Q_P, at most 3 elements are recorded in L_G or L_T.
- For each query of Q_{IE} or Q_{SE}, at most 2 new elements are recorded in L_G or L_T.
- For each query of Q_{PK}, at most 4 new elements are recorded in L_G or L_T.
- For each query of Q_{PR}, at most 2 new elements are recorded in L_G or L_T.
- For each query of Q_D, at most 6 new elements are recorded in L_G or L_T.

Hence, we have

$$|L_G| + |L_T| \leqslant 5 + 3q_O + 2q_{IE} + 2q_{SE} + 4q_{PK} + 2q_{PR} + 6q_D + 1.$$

Let $6 \leqslant 3q_O + 4q_{IE} + 4q_{SE} + 2q_{PK} + 4q_{PR}$, we have

$$|L_G| + |L_T| \leqslant 3q_O + 2q_{IE} + 2q_{SE} + 4q_{PK} + 2q_{PR} + 6q_D + 6 \leqslant 6q.$$

(3) In the following, we discuss the degrees of all multivariate polynomials in $\{P_G\}$.

- All polynomials in $\{S\}$ and $\{T\}$ are of degree 1.
- In Q_{IE}, each polynomial $P_{G,IE,i,k}$ has degree at most 2.
- In Q_{SE}, each polynomial $P_{G,SE,i,1}$ has degree 1.
- In Q_D, each polynomial $P_{G,D,i,k}$ has degree at most 2.
- In Q_G, the polynomial $P_{G,Q,i,3}$ is generated by $P_{G,Q,i,3} = P_{G,Q,i,1} + P_{G,Q,i,2}$. Hence the degree of $P_{G,Q,i,3}$ is less than or equal to the maximal degree of $P_{G,Q,i,1}$ and $P_{G,Q,i,2}$.

Therefore, the degrees of all multivariate polynomials in $\{P_G\}$ are at most 2.

(4) In the following, we obtain that the degrees of all multivariate polynomials in $\{P_T\}$ are at most 4:

- All polynomials in $\{ V \}$ are of degree 1.
- In Q_P, the degree of each polynomial $P_{T,P,i,k}$ is at most 4 since each polynomial P_G has degree at most 2.
- In Q_{SE}, the degree of each polynomial $P_{T,SE,i,2}$ is 2.
- In Q_T, the polynomial $P_{T,Q,i,3}$ is generated by $P_{T,Q,i,3} = P_{T,Q,i,1} + P_{T,Q,i,2}$. Hence, the degree of $P_{G,Q,i,3}$ is less than or equal to the maximal degree of $P_{T,Q,i,1}$ and $P_{T,Q,i,2}$.

In the following, let us discuss the probability that $\mathcal{A}_{NL-I}$ wins the game g_{NL-I}. After completing the game g_{NL-I}, the challenger $\mathcal{B}$ chooses random values x, u_0, u_1, $\{s_1, s_2, \dots,\}$, $t_1, t_2, \dots,$ in Z_q^*, which represent the values X, U_0, U_1, $\{S\}$, $\{T\}$ in the group G. $\mathcal{B}$ also chooses random values $\{v_1, v_2, \dots,\}$ in Z_q^*, which represent the values $\{V\}$ in the group G_T. $\mathcal{A}_{NL-I}$ is said to win the game g_{NL-I} if one of the following cases happens:

- Case 1. There is a collision in G or G_T which can be described as follows:
  (i) In the list L_G, there exist two polynomials $P_{G,i}$ and $P_{G,j}$ such that $P_{G,i}(x\ , u_0, u_1, \{s\}, \{t\}) = P_{G,j}(x, u_0, u_1, \{s\}, \{t\})$.
  (ii) In the list L_T, there exist two polynomials $P_{T,i}$ and $P_{T,j}$ such that $P_{T,i}(x, u_0, u_1, s, \{t\}, \{v\}) = P_{T,j}(x, u_0, u_1, s, \{t\}, \{v\})$.
- Case 2. The adversary $\mathcal{A}_{NL-I}$ outputs $\beta' = \beta$ in the *Guess* phase.

In the real CL-IND-CCA game, the success probability in the simulated game g_{NL-I} is an upper bound of the success probability of $\mathcal{A}_{NL-I}$. Let us discuss the probabilities of two cases as follows.

- Case 1. If $\mathcal{A}_{NL-I}$ can find any collisions within G or G_T, one can solve the discrete logarithm problem in G or G_T. Let $P_{G,i}$ and $P_{G,j}$ denote two distinct polynomials in L_G. Then we obtain the polynomials $P_{G,C} = P_{G,i} - P_{G,j}$ is a non-zero polynomial of degree at most 2. By applying Lemma 2 in Section 2 with $\lambda = 0$, the probability that $P_{G,C}(x, u_0, u_1, \{s\}, \{t\}) = 0$ in Z_q is at most $\frac{2}{p}$. Since there are $\binom{|L_G|}{2}$ different pairs $(P_{G,i}, P_{G,j})$, the probability that Case 1 occurs is at most $\frac{2}{p}\binom{|L_G|}{2}$. Similarly, the collision probability in L_T is at most $\frac{4}{p}\binom{|L_T|}{2}$ since the polynomials in L_T have degree at most 4.
- Case 2. If $\mathcal{A}_{NL-I}$ can't find any collision in G or G_T, the view of $\mathcal{A}_{NL-I}$ in the game g_{NL-I} is identical to that in the real CL-IND-CCA game. If the adversary $\mathcal{A}_{NL-I}$ doesn't obtain any useful information in the game g_{NL-I}, she/he still has the probability $\frac{1}{2}$ on average to output a correct $\beta' = \beta$.

Now we evaluate the probability that $\mathcal{A}_{NL-I}$ wins the game g_{NL-I}, denoted by Pr_{NL-I}. Firstly, we define two events of Pr_{NL-I} as follows.

(1) The event *FAC* denotes that $\mathcal{A}_{NL-I}$ can find a collision in G or G_T.
(2) The event *GBC* denotes that $\mathcal{A}_{NL-I}$ can output $\beta' = \beta$.

Meanwhile, let $\overline{FAC}$ and $\overline{GBC}$ denote the complement events of *FAC* and *GBC*, respectively. The probability that $\mathcal{A}_{NL-I}$ wins g_{NL-I} can be bounded by

$$Pr_{NL-I} \leqslant Pr[FAC] + Pr[\overline{FAC} \wedge GBC].$$

Here, as discussed in Case 1, the probabilities that $\mathcal{A}_{NL-I}$ can find a collision in G and G_T are $\frac{2}{p}\ \binom{|L_G|}{2}$ and $\frac{4}{p}\binom{|L_T|}{2}$, respectively. Hence, we have

$$Pr[FAC] \leqslant \left[\frac{2}{p}\binom{|L_G|}{2} + \frac{4}{p}\binom{|L_T|}{2}\right] \leqslant \frac{2}{p}(|L_G| + |L_T|)^2 \leqslant \frac{72q^2}{p}.$$

On the other hand, in case $\mathcal{A}_{NL-I}$ can't find collisions in G or G_T, $\mathcal{A}_{NL-I}$ still has probability $\frac{1}{2}$ on average to make a correct guess of β'. Therefore, we have

$$Pr_{NL-I} \leqslant Pr[FAC] + Pr[\overline{FAC} \wedge GBC] \leqslant \frac{72q^2}{p} + \left(1 - \frac{72q^2}{p}\right) \cdot \frac{1}{2}.$$

The advantage of $\mathcal{A}_{NL-I}$ is

$$Adv_A \leqslant \left| \frac{72q^2}{p} + \frac{1}{2} \cdot \left(1 - \frac{72q^2}{p}\right) - \frac{1}{2} \right| = \frac{36q^2}{p},$$

which is negligible if $q = poly(\log p)$. □

Theorem 2. *In the generic bilinear group model, the non-leakage version Π_{NL} of our LR-CL-KE scheme is CL-IND-CCA secure against Type II adversary* (*honest-but-curious KGC*)*.*

Proof. Let $\mathcal{A}_{NL-II}$ be a Type II adversary who can break the non-leakage CL-KE scheme Π_{NL}. Meanwhile, the adversary $\mathcal{A}_{NL-II}$ is allowed to issue all queries at most q times. The advantage that $\mathcal{A}_{NL-II}$ breaks Π_{NL} is bounded by the success probability of $\mathcal{A}_{NL-II}$ in the game g_{NL-II} which is played by both the adversary $\mathcal{A}_{NL-II}$ and a challenger $\mathcal{B}$ as follows:

Game g_{NL-II}: In the game g_{NL-II}, there are four phases, namely, *Setup*, *Phase* 1, *Challenge* and *Guess*.

- *Setup*: In this phase, the challenger $\mathcal{B}$ constructs two lists L_G and L_T to record the elements in G and G_T, respectively. $\mathcal{B}$ also maintains two lists L_{IK} and L_{SK} to record the user's initial key and the user's secret key, respectively. The forms of L_G, L_T, L_{IK} and L_{SK} are the same with those described in the game g_{NL-I}. At the end of this phase, the challenger $\mathcal{B}$ sends the bit strings of the public parameters *PP* to $\mathcal{A}_{NL-II}$. Since $\mathcal{A}_{NL-II}$ represents an honest-but-curious KGC, $\mathcal{B}$ also sends the system secret key X (using the form of bit string) to $\mathcal{A}_{NL-II}$.
- *Phase* 1: Since $\mathcal{A}_{NL-II}$ models the honest-but-curious KGC, $\mathcal{A}_{NL-II}$ can obtain the user's initial key by issuing the queries Q_G, Q_T and Q_P. Meanwhile, $\mathcal{A}_{NL-II}$ is not allowed to perform the Public key replacement query. In this phase, $\mathcal{A}_{NL-II}$ can adaptively issue the queries as follows:
  - *Group G query* $Q_G(\phi_{G,Q,i,1}, \phi_{G,Q,i,2}, operation)$: The query is identical to Q_G presented in the game g_{NL-I}.
  - *Group G_T query* $Q_T(\phi_{T,Q,i,1}, \phi_{T,Q,i,2}, operation)$: The query is identical to Q_T presented in the game g_{NL-I}.
  - *Pairing query* $Q_P(\phi_{G,P,i,1}, \phi_{G,P,i,2})$: The query is identical to Q_P presented in the game g_{NL-I}.
  - *Secret key extract query* $Q_{SE}(ID_{SE,i})$: The query is identical to Q_{SE} presented in the game g_{NL-I}.

- *Public key retrieve query* $Q_{PK}(ID_{PK,i})$: For the i-th *Public key retrieve query* with an identity $ID_{PK,i}$, $\mathcal{B}$ runs three steps as follows:
  (i) $\mathcal{B}$ checks whether $ID_{PK,i}$ was recorded in L_{IK}. If so, $\mathcal{B}$ may obtain the corresponding polynomial of *QID* for $ID_{PK,i}$. Otherwise, $\mathcal{B}$ uses the records of the queries Q_G, Q_T and Q_P to obtain the corresponding polynomials of *DID* and *QID* for $ID_{PK,i}$ while updating the list L_{IK} for $ID_{PK,i}$.
  (ii) $\mathcal{B}$ checks whether $ID_{PK,i}$ was recorded in L_{SK}. If so, $\mathcal{B}$ may obtain the corresponding polynomial of *RID* for $ID_{PK,i}$. Otherwise, $\mathcal{B}$ may issue the *Secret key extract query* $Q_{SE}(ID_{PK,i})$ to obtain the corresponding polynomial of *RID* for $ID_{PK,i}$.
  (iii) Finally, $\mathcal{B}$ returns *QID* and *RID* (with the form of bit strings) by searching the lists L_G and L_T, respectively.
- *Decrypt query* $Q_D(ID_{D,i}, C_i, CT_i)$: For the i-th *Decrypt* round, when $\mathcal{A}_{NL-II}$ queries $\mathcal{B}$ along with a user's identity $ID_{D,i}$ and a ciphertext pair (C_i, CT_i), $\mathcal{B}$ performs the following two parts to obtain the encryption key K:
  (1) $\mathcal{B}$ first obtains the user's initial key *DID* and secret key *SID* from the lists L_{IK} and L_{SK} as follows:
    (i) $\mathcal{B}$ checks whether the user's initial key *DID* of $ID_{D,i}$ has been recorded in L_{IK}. If so, $\mathcal{B}$ obtains the corresponding polynomial of *DID* for $ID_{D,i}$ in L_{IK}. Otherwise, $\mathcal{B}$ uses the records of the queries Q_G, Q_T and Q_P to obtain the corresponding polynomials of *DID* and *QID* for $ID_{D,i}$ while updating the list L_{IK} for $ID_{D,i}$.
    (ii) $\mathcal{B}$ uses $ID_{D,i}$ to find the user's secret key *SID* in the list L_{SK}. If so, $\mathcal{B}$ obtains *SID* in L_{SK}. Otherwise, $\mathcal{B}$ issues the query $Q_{SE}(ID_{D,i})$ to obtain *SID*.
    (iii) Hence, $\mathcal{B}$ have obtained the corresponding polynomials $P_{G,IE,k,1}$ and $P_{G,SE,l,1}$, which represent *DID* and *SID*, respectively.
  (2) $\mathcal{B}$ can obtain the encryption key K by using the same steps in the *Decrypt query* of the game g_{NL-I}.
  Finally, $\mathcal{B}$ computes the bit string $\phi_{T,D,i,4} = \phi_{T,D,i,1} \oplus \phi_{T,D,i,2}$ (which denotes K). At the end of this query, $\mathcal{B}$ obtains the plaintext $msg_i = D_K(CT_i)$ by using the decryption key K. Finally, $\mathcal{B}$ returns msg_i to $\mathcal{A}_{NL-II}$.

- *Challenge*: This phase is similar to the *Challenge phase* described in g_{NL-I}. The only difference is that ID^* is not allowed to be queried in the *Secret key extract query* of *Phase* 1 since $\mathcal{A}_{NL-II}$ is the honest-but-curious KGC.
- *Guess*: The adversary $\mathcal{A}_{NL-II}$ outputs $\beta' \in \{0, 1\}$. If $\beta' = \beta$, we say that the adversary $\mathcal{A}_{NL-II}$ wins the game g_{NL-II}.

As the same arguments in Theorem 1, we can compute the success probability of $\mathcal{A}_{NL-II}$ in the game g_{NL-II}. We first compute the number of $|L_G| + |L_T|$. We have $|L_G| + |L_T| \leqslant 5 + 3q_O + 2q_{SE} + 4q_{PK} + 4q_D + 1 = 3q_O + 2q_{SE} + 4q_{PK} + 4q_D + 6 \leqslant 4q$ by letting $6 \leqslant q_O + 2q_{SE}$. And we may obtain $Pr[FAC] \leqslant \frac{32q^2}{p}$, where the event *FAC* denotes that $\mathcal{A}_{NL-II}$ can find a collision in G or G_T. So, the success probability of $\mathcal{A}_{NL-II}$ is

$$Pr_{NL-II} \leqslant Pr[FAC] + Pr[\overline{FAC} \wedge GBC] \leqslant \frac{32q^2}{p} + \frac{1}{2} \cdot \left(1 - \frac{32q^2}{p}\right).$$

Hence, the adversary $\mathcal{A}_{NL-II}$'s advantage is

$$Adv_A \leqslant \left|\frac{32q^2}{p} + \frac{1}{2} \cdot \left(1 - \frac{32q^2}{p}\right) - 1/2\right| = \frac{16q^2}{p},$$

which is negligible if $q = poly(\log p)$. □

Theorem 3. *In the generic bilinear group model, the proposed LR-CL-KE scheme* Π *is LR-CL-IND-CCA secure against Type I adversary (outsider) under the continual leakage model.*

Proof. In Theorem 1, we have shown that the non-leakage version Π_{NL} of the proposed LR-CL-KE scheme is CL-IND-CCA secure against Type I adversary. Under the continual leakage model, an adversary is allowed to issue two additional leakage queries, *Initial key extract leak query* and *Decrypt leak query*. Let $\mathcal{A}_{LR-I}$ be a Type I adversary who may break the proposed LR-CL-KE scheme Π_{LR}. $\mathcal{A}_{LR-I}$ can adaptively issue the queries at most q times in total. In the following, we present a game g_{LR-I} extended from the game g_{NL-I} in Theorem 1 as follows. **Game** g_{LR-I}: In the game g_{LR-I}, there are four phases that include *Setup*, *Phase* 1, *Challenge* and *Guess*. Four phases are presented as follows:

- *Setup*: The phase is identical to that of the game g_{NL-I}.
- *Phase* 1: In this phase, the adversary $\mathcal{A}_{LR-I}$ can issue two additional leakage queries than the adversary $\mathcal{A}_{NL-I}$ in the game g_{NL-I}, namely, *Initial key extract leak query* and *Decrypt leak query*. In order to record the leakage information for two kinds of leak queries, we build four initial-empty lists $L_{f,IE}$, $L_{h,IE}$, $L_{f,D}$ and $L_{h,D}$ as follows:

$$L_{f,IE} = \{(f_{IE,i}, \Lambda f_{IE,i}), 1 \leqslant i \leqslant q_{IE}\},$$
$$L_{h,IE} = \{(h_{IE,i}, \Lambda h_{IE,i}), 1 \leqslant i \leqslant q_{IE}\},$$
$$L_{f,D} = \{(f_{D,j}, \Lambda f_{D,j}), 1 \leqslant j \leqslant q_D\},$$
$$L_{h,D} = \{(h_{D,j}, \Lambda h_{D,j}), 1 \leqslant j \leqslant q_D\}.$$

Two leakage functions $f_{IE,i}$ and $h_{IE,i}$ are, respectively, used to model the adversary's leak ability for two sub-algorithms *Extract-1* and *Extract-2* of the i-th *Initial key extract* round. Also, two leakage functions $f_{S,j}$ and $h_{S,j}$ are, respectively, used to model the adversary's leak ability for two sub-algorithms *Decrypt-1* and *Decrypt-2* of a user's j-th Decrypt round. Moreover, the leakage information $\Lambda f_{IE,i}$, $\Lambda h_{IE,i}$, $\Lambda f_{D,j}$ and $\Lambda h_{D,j}$ denote the outputs of four leakage functions $f_{IE,i}$, $h_{IE,i}$, $f_{D,j}$ and $h_{D,j}$, respectively. In the following, we describe two additional leakage queries as follows:

  - *Initial key extract leak query* $(f_{IE,i}, h_{IE,i}, i)$: For the i-th *Initial key extract query*, $\mathcal{A}_{LR-I}$ can issue the *Initial key extract leak query* only once by providing two leakage functions $f_{IE,i}$ and $h_{IE,i}$ such that $|f_{IE,i}| \leqslant \lambda$ and $|h_{IE,i}| \leqslant \lambda$. $\mathcal{B}$ computes and sends the leakage information $(\Lambda f_{IE,i}, \Lambda h_{IE,i})$ to $\mathcal{A}_{LR-I}$, where $\Lambda f_{IE,i} = f_{IE,i}(SK_{i-1,1}, \gamma_i, a_i)$ and $\Lambda h_{IE,i} = h_{IE,i}(SK_{i-1,2}, TI_{IE}, a_i)$. Meanwhile, $\mathcal{B}$ records $(f_{IE,i}, \Lambda f_{IE,i})$ in the list $L_{f,IE}$ and $(h_{IE,i}, \Lambda h_{IE,i})$ in the list $L_{h,IE}$.
  - *Decrypt leak query* $(f_{D,j}, h_{D,j}, j)$: For the j-th *Decrypt query*, $\mathcal{A}_{LR-I}$ can issue the *Decrypt leak query* only once by providing two leakage functions $f_{D,j}$ and $h_{D,j}$ such that $|f_{D,i}| \leqslant \lambda$ and $|h_{D,i}| \leqslant \lambda$. $\mathcal{B}$ computes and sends the leakage information $(\Lambda f_{D,j}, \Lambda h_{D,j})$ to $\mathcal{A}_{LR-I}$, where $\Lambda f_{D,j} = f_{D,j}(DID_{j-1,1}, b_j, c_j)$ and $\Lambda h_{D,j} = h_{D,j}(DID_{j-1,2}, TI_{1,j}, TI_{2,j}, b_j, c_j, K_{1,j}, K_{2,j}, K_j)$. Meanwhile, $\mathcal{B}$ records $(f_{D,j}, \Lambda f_{D,j})$ in the list $L_{f,D}$ and $(h_{D,j}, \Lambda h_{D,j})$ in the list $L_{h,D}$.
- *Challenge*: The adversary $\mathcal{A}_{LR-I}$ gives a target identity ID^* and a plaintext pair (msg_0^*, msg_1^*) to $\mathcal{B}$. This phase is identical to the *Challenge* phase in g_{NL-I}. Finally, $\mathcal{B}$ sends C^* and CT^* to $\mathcal{A}_{LR-I}$.
- *Guess*: The adversary $\mathcal{A}_{LR-I}$ outputs $\beta' \in \{0, 1\}$. If $\beta' = \beta$, we say that the adversary $\mathcal{A}_{LR-I}$ wins the game g_{LR-I}.

In the game g_{LR-I}, $\mathcal{A}_{LR-I}$ has higher probability to cause collisions by making use of the leakage functions. Two leakage information $\Lambda f_{IE,i}$ and $\Lambda h_{IE,i}$, respectively, respresent the ouputs of two leakage functions $f_{IE,i}$ and $h_{IE,i}$ in the i-th *Initial key extract query*. By $\Lambda f_{IE,i}$ and $\Lambda h_{IE,i}$, the leaked information about $(SK_{i-1,1}, \gamma_i, a_i)$ and $(SK_{i-1,2}, TI_{IE}, a_i)$ are discussed below:

- γ_i: The value γ_i is used to generate the initial key (DID_0, QID) for $ID_{IE,i}$ in the *Initial key extract query*. By Definition 2 in Section 3.2, if $ID_{IE,i}$ has been queried in the *Initial key extract query*, it is not allowed to be a target identity in the *Challenge* phase. Hence, the leakage of γ_i is useless for $\mathcal{A}_{LR-I}$.
- $(SK_{i-1,1}, SK_{i-1,2})$: Since the system secret key $X = SK_{i-1,1} \cdot SK_{i-1,2}$, obtaining some leakage information of $SK_{i,1}$ and $SK_{i,2}$ is contributive to learn partial information of X for $\mathcal{A}_{LR-I}$. Indeed, $\mathcal{A}_{LR-I}$ can learn at most 2λ bits of the system secret key X.
- a_i: The parameter a_i is used to generate the next system secret key $(SK_{i,1}, SK_{i,2})$ from $(SK_{i-1,1}, SK_{i-1,2})$. Hence, $\mathcal{A}_{LR-I}$ may obtain at most λ bits of $SK_{i,1}$ and $SK_{i,2}$, respectively.
- TI_{IE}: The temporary information TI_{IE} is only used to generate the initial key DID_0 for $ID_{IE,i}$. TI_{IE} is helpless in this game g_{LR-I} since $ID_{IE,i}$ is not allowed to be a target identity in the *Challenge* phase.

On the other hand, two leakage information $\Lambda f_{D,j}$ and $\Lambda h_{D,j}$, respectively, respresent the ouputs of two leakage functions $f_{D,j}$ and $h_{D,j}$ in the j-th *Decrypt leak query*. By $\Lambda f_{D,j}$ and $\Lambda h_{D,j}$, the leaked information about $(DID_{j-1,1}, b_j, c_j)$ and $(DID_{j-1,2}, TI_1, j, TI_2, j, b_j, c_j, K_{1,j}, K_{2,j}, K_j)$ are discussed below:

- $(DID_{j-1,1}, DID_{j-1,2})$: Since the user's first initial key $DID_0 = DID_{j-1,1}, DID_{j-1,2}$, obtaining some leakage information of $DID_{j-1,1}$ and $DID_{j-1,2}$ is contributive to learn partial information of DID_0 for $\mathcal{A}_{LR-I}$. Indeed, $\mathcal{A}_{LR-I}$ can learn at most 2λ bits of the user's initial key DID_0.

- $(TI_{1,j}, TI_{2,j})$: The temporary information $TI_{1,j}$ and $TI_{2,j}$ are used to compute $K_{1,j}$ and $K_{2,j}$, respectively. Since each encryption key $K_j = K_{1,j} \oplus K_{2,j}$ is independent with each other, obtaining $TI_{1,j}$ and $TI_{2,j}$ is helpless in the *Guess* phase.
- b_j: The parameter b_j is used to compute the user's initial key $(DID_{j,1}, DID_{j,2})$ from $(DID_{j-1,1}, DID_{j-1,2})$. Therefore, $\mathcal{A}_{LR-I}$ can learn at most λ bits of $DID_{j,1}$ and $DID_{j,2}$, respectively.
- c_j: The parameter c_j is used to compute the user secret key $(SID_{j,1}, SID_{j,2})$ from $(SID_{j-1,1}, SID_{j-1,2})$. Therefore, $\mathcal{A}_{LR-I}$ can learn at most λ bits of $SID_{j,1}$ and $SID_{j,2}$, respectively.
- $(K_{1,j}, K_{2,j}, K_j)$: For the j-th *Decrypt query*, $\mathcal{A}_{LR-I}$ can use the leakage function $h_{D,j}$ to obtain the leakage information about $(K_{1,j}, K_{2,j}, K_j)$ once for totally at most λ bits. Since $K_{1,j}$ and $K_{2,j}$ can only be used to generate K_j, adversary $\mathcal{A}_{LR-I}$ can learn at most λ bits information about K_j in the game g_{L-IR}.

Now, let us discuss the success probability Pr_{LR-I} that $\mathcal{A}_{LR-I}$ wins the game g_{LR-I}. Since $\mathcal{A}_{LR-I}$ can get the secret key of any entity, $\mathcal{A}_{LR-I}$ always outputs a correct β' when she/he gets the target user's initial key DID_0 or the system secret key X. Firstly, we define three events of Pr_{LR-I} as follows.

(1) The event *EI* denotes that the adversary $\mathcal{A}_{LR-I}$ may obtain DID_0 completely from the leakage information $\Lambda f_{D,j}$ and $\Lambda h_{D,j}$.
(2) The event *ES* denotes that the adversary $\mathcal{A}_{LR-I}$ may obtain the system secret key X completely from the leakage information $\Lambda f_{IE,i}$ and $\Lambda h_{IE,i}$.
(3) The event *EC* denotes that the adversary $\mathcal{A}_{LR-I}$ may output a correct β'.

In addition, let $\overline{ES}$ and $\overline{EI}$, respectively, denote the complement events of *ES* and *EI*. The success probability Pr_{LR-I} that the adversary $\mathcal{A}_{LR-I}$ wins the game g_{LR-I} is bounded as follows.

$$\begin{aligned} Pr_{LR-I} &= Pr[EC] \\ &= Pr[EC \wedge ES] + Pr[EC \wedge \overline{ES}] \\ &= Pr[EC \wedge ES] + Pr[EC \wedge \overline{ES} \wedge EI] + Pr[EC \wedge \overline{ES} \wedge \overline{EI}] \\ &= Pr[EC \wedge ES] + Pr[EC \wedge \overline{ES} \wedge EI] + Pr[EC|\overline{ES} \wedge \overline{EI}] \cdot (Pr[\overline{ES} \wedge \overline{EI}]. \end{aligned}$$

Since $Pr[EC \wedge ES] \leqslant Pr[ES]$ and $Pr[EC \wedge \overline{ES} \wedge EI] \leqslant Pr[\overline{ES} \wedge EI]$, we obtain

$$Pr_{LR-I} \leqslant Pr[ES] + Pr[\overline{ES} \wedge EI] + Pr[OBC|\overline{ES} \wedge \overline{EI}] \cdot Pr[\overline{ES} \wedge \overline{EI}].$$

Let us focus on $Pr[EC|\overline{ES} \wedge \overline{EI}]$. Under the condition $\overline{ES} \wedge \overline{EI}$, $\mathcal{A}_{LR-I}$ can't obtain the useful information to output β' correctly. Hence, $Pr[EC|\overline{ES} \wedge \overline{EI}]$ is $\frac{1}{2}$ on average. Thus, we obtain

$$Pr[EC|\overline{ES} \wedge \overline{EI}] \cdot Pr[\overline{ES} \wedge \overline{EI}] = (1/2)\big(1 - Pr[ES] - Pr[\overline{ES} \wedge EI]\big).$$

Hence, we have

$$Pr_{LR-I} \leqslant 1/2 + (1/2)\big(Pr[ES] + Pr[\overline{ES} \wedge EI]\big).$$

Lemmas 3 and 4 below offer upper bounds for $Pr[ES]$ and $Pr[\overline{ES} \wedge EI]$, respectively. By assuming these results, the adversary $\mathcal{A}_{LR-I}$'s advantage is

$$\begin{aligned} Adv_A &\leqslant \left|Pr_{LR-I} - \frac{1}{2}\right| = \left|\frac{1}{2}\big(Pr[ES] + Pr[\overline{ES} \wedge EI]\big)\right| \\ &= \left|\frac{1}{2}\left(O\left(\frac{q^2}{p} \cdot 2^{2\lambda}\right) + O\left(\frac{q^2}{p} \cdot 2^{2\lambda}\right)\right)\right| \leqslant O\left(\frac{q^2}{p} \cdot 2^{2\lambda}\right). \end{aligned}$$

Hence, the advantage of the adversary $\mathcal{A}_{LR-I}$ breaking our LR-CL-KE scheme is $O(\frac{q^2}{p} \cdot 2^{2\lambda})$. By Corollary 1, if $\lambda \ll \frac{\log(p)}{2}$, we say that the proposed scheme Π_{LR} is LR-CL-IND-CCA secure against Type I adversary (outsider) under the continual leakage model. □

Lemma 3. $Pr[ES] \leqslant O(\frac{q^2}{p} \cdot 2^{2\lambda})$.

Proof. In the *Initial key extract* algorithm of our LR-CL-KE scheme, the initial key of a user is a signature on her/his identity *ID*, which is generated by the signature scheme proposed by Galindo and Virek (2013). Hence, the probability $Pr[ES]$ is then bounded by the probability that the adversary can compute the secret key in Galindo and Vivek's scheme. By applying the Lemma 5 in Galindo and Virek (2013), we have $Pr[ES] \leqslant O(\frac{q^2}{p} \cdot 2^{2\lambda})$. □

Lemma 4. $Pr[\overline{ES} \wedge EI] \leqslant O(\frac{q^2}{p} \cdot 2^{2\lambda})$.

Proof. Under the condition $\overline{ES}$, $\mathcal{A}_{LR-I}$ can't obtain the system secret key X. We focus on the probability that $\mathcal{A}_{LR-I}$ can obtain DID_0 completely under the condition $\overline{ES}$. As described earlier, no useful information of DID_0 can be obtained from the leakage information $\Lambda f_{IE,i}$ and $\Lambda h_{IE,i}$ in the *Initial key extract leak query*. However, $\mathcal{A}_{LR-I}$ may obtain some useful information of DID_0 by the *Decrypt leak query*. In such a case, $Pr[\overline{ES} \wedge EI]$ denotes that $\mathcal{A}_{LR-I}$ can obtain the user's initial key without using the leakage functions $f_{IE,i}$ and $h_{IE,i}$. As described earlier, the useful information to generate the user's initial key DID_0 from the leakage functions $f_{D,j}$ and $h_{D,j}$ are $(DID_{j-1,1}, DID_{j-1,2})$ and b_j. In our scheme, the user's initial key is updated in the beginning of two sub-algorithms *Decrypt-1* and *Decrypt-2*. Hence, the adversary can learn at most 2λ bits about DID_0. Considering the advantage that $\mathcal{A}_{NL-I}$ obtains in Theorem 1, the probability that the adversary $\mathcal{A}_{NL-I}$ can find a collision is $Pr[FAC] \leqslant \frac{72q^2}{p}$. By applying Lemma 2, we have $Pr[\overline{ES} \wedge EI]$ is bounded by $\frac{72q^2}{p} \cdot 2^{2\lambda}$. Hence, we obtain $Pr[\overline{ES} \wedge EI] \leqslant O(\frac{q^2}{p} \cdot 2^{2\lambda})$. □

Theorem 4. *In the generic bilinear group model, the proposed LR-CL-KE scheme Π is LR-CL-IND-CCA secure against Type II adversary (honest-but-curious KGC) under the continual leakage model.*

Proof. In Theorem 2, we have shown that the non-leakage version Π_{NL} of the proposed LR-CL-KE scheme is CL-IND-CCA secure against Type II adversary. Under the continual leakage model, an adversary is allowed to issue two additional leakage queries, *Initial key extract leak query* and *Decrypt leak query*. Let $\mathcal{A}_{LR-II}$ be a Type II adversary who may break the proposed LR-CL-KE scheme Π_{LR}. $\mathcal{A}_{LR-II}$ can adaptively issue the queries at most q times in total. In the following, we present a game g_{LR-II} extended from the game g_{NL-II} in Theorem 2 as follows. **Game** g_{LR-II}**.** In the game g_{LR-II}, there are four phases, *Setup*, *Phase* 1, *Challenge* and *Guess*.

- *Setup*: The phase is identical to that of g_{NL-II}.
- *Phase* 1: In this phase, $\mathcal{A}_{LR-II}$ can issue an extra leakage query (i.e. *Decrypt leak query*) than the adversary $\mathcal{A}_{NL-II}$ in the game g_{NL-II}. In order to record the leakage information for the *Decrypt leak query*, we build two initial-empty lists $L_{f,D}$ and $L_{h,D}$, which are identical to those in the game g_{LR-I}.
  - *Decrypt leak query* $(f_{D,j}, h_{D,j}, j)$: This query is identical to the *Decrypt leak query* described in g_{LR-I}.
- *Challenge*: The adversary $\mathcal{A}_{LR-II}$ gives a target identity ID^* and a plaintext pair (msg_0^*, msg_1^*) to $\mathcal{B}$. This phase is identical to the *Challenge* phase in g_{NL-II}. Finally, $\mathcal{B}$ sends C^* and CT^* to $\mathcal{A}_{LR-II}$.
- *Guess*: The adversary $\mathcal{A}_{LR-II}$ outputs $\beta' \in \{0,1\}$. If $\beta' = \beta$, we say that the adversary $\mathcal{A}_{LR-II}$ wins the game g_{LR-II}.

In g_{LR-II}, $\mathcal{A}_{LR-II}$ has higher probability to cause collisions by making use of the leakage functions. In the j-th *Decrypt leak query*, two leakage information $\Lambda f_{D,j}$ and $\Lambda h_{D,j}$, respectively, respresent the ouputs of two leakage functions $f_{D,j}$ and $h_{D,j}$. By $\Lambda f_{D,j}$ and $\Lambda h_{D,j}$, the adversary $\mathcal{A}_{LR-II}$ can obtain the partial information of both $(SID_{j-1,1}, b_j, c_j)$ and $(SID_{j-1,2}, TI_1, i, TI_2, i, b_j, c_j, K_{1,i}, K_{2,i}, K_i)$. The discussions on the partial leakage information of $(TI_{1,i}, TI_{2,i}, b_j, c_j, K_{1,i}, K_{2,i}, K_i)$ are the same with those in Theorem 3. In addition, the leaked information about $(SID_{j-1,1}, SID_{j-1,2})$ is discussed below:

- $(SID_{j-1,1}, SID_{j-1,2})$: Since the user's secret key $SID_0 = SID_{j-1,1} \cdot SID_{j-1,2}$, obtaining some leakage information of $SID_{j-1,1}$ and $SID_{j-1,2}$ is contributive to learn the partial information of SID_0 for $\mathcal{A}_{LR-II}$. Indeed, $\mathcal{A}_{LR-II}$ can learn at most 2λ bits of the user's secret key SID_0.

Now, let us discuss the success probability Pr_{LR-II} that the adversary $\mathcal{A}_{LR-II}$ wins the game g_{LR-II}. Since $\mathcal{A}_{LR-II}$ holds the system secret key X, $\mathcal{A}_{LR-II}$ can obtain each user's initial key DID_0. If $\mathcal{A}_{LR-II}$ can obtain the user's secret key SID, $\mathcal{A}_{LR-II}$ always outputs a correct β'. Here we define two events of Pr_{LR-II} as follows.

(1) The event *EU* denotes that the user's secret key SID_0 can be obtained completely by $\mathcal{A}_{LR-II}$ from the leakage information $\Lambda f_{D,j}$ and $\Lambda h_{D,j}$.
(2) The event *EC* denotes that $\mathcal{A}_{LR-II}$ can guess β' correctly.

In addition, the event $\overline{EU}$ is the complement event of EU. The success probability Pr_{LR-II} that the adversary $\mathcal{A}_{LR-II}$ wins the game g_{LR-II} is bounded as follows.

$$\begin{aligned} Pr_{LR-II} &= Pr[EC] = Pr[EC \wedge EU] + Pr[EC \wedge \overline{EU}] \\ &= Pr[EC \wedge EU] + Pr[EC|\overline{EU}] \cdot Pr[\overline{EU}]. \end{aligned}$$

Since $Pr[EC \wedge EU] \leqslant Pr[EU]$, we have

$$Pr_{LR-II} \leqslant Pr[EU] + Pr[EC|\overline{EU}] \cdot Pr[\overline{EU}].$$

Let us focus on $Pr[EC|\overline{EU}]$. Under the condition $\overline{EU}$, $\mathcal{A}_{LR-II}$ can't obtain useful information to output β' correctly. Hence, $Pr[EC|\overline{EU}]$ is equal to $\frac{1}{2}$ plus the advantage $O(\frac{q^2}{p})$ of the adversary $\mathcal{A}_{NL-II}$ in Theorem 2. Thus, we obtain

$$Pr[EC|\overline{EU}] \cdot Pr[\overline{EU}] = \frac{1}{2}\big(1 - Pr[EC \wedge EU]\big).$$

Hence, we have $Pr_{LR-I} \leqslant \frac{1}{2} + \frac{1}{2}Pr[EU]$. By assuming Lemma 5 below, we obtain an upper bound for $\mathcal{A}_{LR-II}$'s advantage as

$$Adv_A \leqslant \left|Pr_{LR-I} - \frac{1}{2}\right| = \left|\frac{1}{2}Pr[EU]\right| \leqslant O\left(\frac{q^2}{p}2^{2\lambda}\right).$$

Thus, the advantage of the adversary $\mathcal{A}_{LR-II}$ breaking our LR-CL-KE scheme is $O(\frac{1}{p}2^{2\lambda})$. By Corollary 1, if $\lambda \ll \frac{\log(p)}{2}$, we say that the proposed scheme Π_{LR} is LR-CL-IND-CCA secure against Type II adversary (honest-but-curious KGC) under the continual leakage model. □

Lemma 5. $Pr[EU] \leqslant O(\frac{q^2}{p}2^{2\lambda})$.

Proof. Considering the advantage that $\mathcal{A}_{NL-II}$ obtains in Theorem 2, the probability that $\mathcal{A}_{NL-II}$ can find a collision is $Pr[FAC] \leqslant \frac{32q^2}{p}$. Since $\mathcal{A}_{LR-II}$ can learn at most 2λ bits information for the user current secret key in the *Decrypt leak query*, by applying Lemma 2, we have Pr[EU] is bounded by $\frac{32q^2}{p}2^{2\lambda}$. Hence, we obtain $Pr[EU] \leqslant O(\frac{q^2}{p})2^{2\lambda})$. □

6. Performance Analysis

In this section, we compare the proposed LR-CL-KE scheme with the leakage-resilient certificateless public key encryption (LR-CL-PKE) scheme proposed by Xiong *et al.* (2013). In the following, we define several notations to analyse the computational costs.

- T_e: The time of executing an exponentiation operation in G or G_T.
- T_p: The time of executing a bilinear pairing operation e: $G \times G \rightarrow G_T$.

Table 1
Comparisons between our LR-CL-KE scheme and the previously proposed schemes.

	The LR-CL-PKE scheme (Xiong *et al.*, 2013)	The proposed LR-CL-KE scheme
Encryption cost	$(n+4)T_e$	$4T_e+4T_p$
Decryption cost	$(n+2)T_p$	$4T_e+4T_p$
Security model	Standard model (Dual system)	GBG model
Security property	LR-CCA1	LR-CCA1
Leakage model	Bounded leakage	Continue leakage

When compared to T_e and T_p, the multiplication operation in the multiplicative group G or G_T is trivial and negligible (Scott, 2011). Table 1 lists the comparisons between the proposed LR-CL-KE scheme and Xiong *et al.*'s LR-CL-PKE scheme (Xiong *et al.*, 2013) in terms of the size of encryption cost, decryption cost, security model, security property and leakage model. Note that a user's private key in Xiong *et al.*'s LR-CL-PKE scheme is a vector with n elements. For the costs of encryption and decryption, Xiong *et al.*'s scheme requires $(n+4)T_e$ and $(n+2)T_p$, respectively. In the proposed LR-CL-KE scheme, $4T_e + T_p$ and $4T_e + 4T_p$ are required for encryption and decryption, respectively.

For the security model and security property, Xiong *et al.* employed the dual system encryption technique (Lewko *et al.*, 2011) to define semi-functional (SF) keys and ciphertexts. In the standard model, they then proved that their scheme possesses the LR-CCA1 security under the bounded leakage model. As mentioned earlier, we formally proved that, in the GBG model, our LR-CL-KE scheme is LR-CCA1 secure against both Type I and Type II adversaries under continual leakage model.

7. Conclusions and Future Work

The first LR-CL-KE scheme under the continual leakage model was proposed in the article. We defined a new adversary model for LR-CL-KE schemes under the continual leakage model. The adversary model also consists of two types of adversaries. Type I adversary can obtain partial information of a user's initial key in the *Decrypt* phase and KGC's system secret key in the *Initial key extract* phase. Type II adversary can obtain partial information of a user's secret key in the *Decrypt* phase since she/he already knows the initial key of any user. In the GBG model, we formally proved that our LR-CL-KE scheme is semantically secure against chosen ciphertext attacks for both Type I and Type II adversaries. It is worth mentioning that the proposed LR-CL-KE scheme achieves only the LR-CCA1 security, but not the LR-CCA2 security. Indeed, it is an interesting and open problem to propose a LR-CCA2 secure LR-CL-PKE or LR-CL-KE scheme under the continual leakage model. Furthermore, up to date, there does not exist leakage-resilient RSA-based certificateless encryption/signature schemes under continual leakage model. Indeed, it is also an interesting issue to design efficient leakage-resilient RSA-based certificateless encryption/signature schemes.

Acknowledgements. The authors would like to appreciate anonymous referees for their valuable comments and constructive suggestions. This research was partially supported by Ministry of Science and Technology, Taiwan, under contract no. MOST106-2221-E-018-007-MY2.

References

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.

Biham, E., Carmeli, Y., Shamir, A. (2008). Bug attacks. In: *CRYPTO'08*, *LNCS*, Vol. 5157, pp. 221–240.

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

Boneh, D., Demillo, R.A., Lipton, R.J. (1997). On the importance of checking cryptographic protocols for faults. In: *EUROCRYPT'97*, *LNCS*, Vol. 1233, pp. 37–51.

Boneh, D., Boyen, X., Goh, E.J. (2005). Hierarchical identity-based encryption with constant size ciphertext. In: *EUROCRYPT'05*, *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.

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

Dodis, Y., Haralambiev, K. (2010). Cryptography against continuous memory attacks. In: *51st Annual IEEE Symposium on Foundations of Computer Science*. IEEE Press, pp. 511–520.

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.

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.

Hu, B., Wong, D., Zhang, Z., Deng, X. (2007). Certificateless signature: a new security model and an improved generic construction. *Designs, Codes and Cryptography*, 42(2), 109–126.

Huang, X., Mu, Y., Susilo, W., Wong, D., Wu, W. (2007). Certificateless signature revisited. In: *ACISP'06*, *LNCS*, Vol. 4586, pp. 308–322.

Hung, Y.H., Huang, S.S., Tseng, Y.M., Tsai, T.T. (2015). Certificateless signature with strong unforgeability in the standard model. *Informatica*, 26(4), 663–684.

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.

Hung, Y.H., Huang, S.S., Tseng, Y.M., Tsai, T.T. (2017) Efficient anonymous multireceiver certificateless encryption. *IEEE Systems Journal*, 11(4), pp. 2602–2613.

Hwang, Y.H., Liu J.K., Chow, S.S.M. (2008). Certificateless public key encryption secure against malicious KGC attacks in the standard model. *Journal of Universal Computer Science*, 14(3), 463–480.

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.

Lewko, A. B., Rouselakis, Y., Waters, B. (2011). Achieving leakage resilience through dual system encryption. In: *TCC'11*, *LNCS*, Vol. 6597, pp. 70–88.

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

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.

Libert, B., Quisquater, J.J. (2006). On constructing certificateless cryptosystems from identity based encryption. In: *PKC'06*, *LNCS*, Vol. 3958, pp. 474–490.

Lin, X.J., Sun, L., Qu, H. (2017). An efficient RSA-based certificateless public key encryption scheme. *Discrete Applied Mathematics*. In press, doi:doi.org/10.1016/j.dam.2017.02.019.

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.

Maurer, U., Wolf, S. (1998). Lower bounds on generic algorithms in groups. In: *EUROCRYPT'98*, *LNCS*, Vol. 1403, pp. 72–84.

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.

Schwartz, J.T. (1980). Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4), 701–717.

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

Shamir, A. (1984). Identity-based cryptosystems and signature schemes. In: *CRYPTO'84*, *LNCS*, Vol. 196, pp. 47–53.

Sharma, G., Bala, S., Verma A. (2016) An improved RSA-based certificateless signature scheme for wireless sensor networks. *International Journal of Network Security*, 18(1), 82–89.

Shoup, V. (1997). Lower bounds for discrete logarithms and related problems. In: *EUROCRYPT'97*, *LNCS*, Vol. 1233, pp. 256–266.

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.

Waters, B. (2005). Efficient identity-based encryption without random oracles. In: *EUROCRYPT'05*, *LNCS*, Vol. 3494, pp. 114–127.

Wu, J.D., Tseng, Y.M., Huang, S.S. (2016). Leakage-resilient ID-based signature scheme in the generic bilinear group model. *Security and Communication Networks*, 9(17), 3987–4001.

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.

Zhang, J. Mao, J. (2012) An efficient RSA-based certificateless signature scheme. *Journal of Systems and Software*, 85(3), 638–642.

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.

Zippel, R. (1979). Probabilistic algorithms for sparse polynomials. In: *EUROSAM'79*, *LNCS*, Vol. 72, pp. 216–226.

J.-D. Wu received the BS degree from the Department of Mathematics, National Changhua University of Education, Taiwan, in 2006. He received the MS degree from the Department of Mathematics, National Changhua University of Education, Taiwan, in 2008. He is currently a PhD candidate in the Department of Mathematics, National Changhua University of Education, Taiwan. His research interests include applied cryptography and pairing-based cryptography.

Y.-M. Tseng is currently 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). In 2006, his paper received the Wilkes Award from The British Computer Society. He has published over one hundred scientific journals and conference papers on various research areas of cryptography, security and computer network. His research interests include cryptography, network security, computer network and mobile communications. 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 network security. He received his PhD from the University of Illinois at Urbana-Champaign in 1997 under the supervision of Professor Bruce C. Berndt.

W.-C. Chou received the BS degree from the Department of Mathematics, National Changhua University of Education, Taiwan, in 2015. He received the MS degree from the Department of Mathematics, National Changhua University of Education, Taiwan, in 2017. His research interests include leakage-resilient cryptography and network security.