INFORMATICA, 2025, Vol. 36, No. 2, 429–452

DOI: https://doi.org/10.15388/25-INFOR597

Leakage-Resilient Public Key Signcryption with Equality Test and Its Application

Tung-Tso TSAI

Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung 202, Taiwan
e-mail: tttsai@mail.ntou.edu.tw

Received: February 2024; accepted: May 2025

Abstract. A public key signcryption with equality test (PKSCET) scheme is a public key signcryption (PKSC) scheme with the property of equality test. However, all the existing PKSCET schemes are vulnerable to a new kind of security threats, called side-channel attacks, which could potentially lead to the unauthorized disclosure of sensitive information or even the compromise of secret keys, undermining the overall confidentiality and integrity of the system. Therefore, this study aims to propose the *first* leakage-resilient PKSCET (LR-PKSCET) scheme that achieves resistance to side-channel attacks. Moreover, the proposed LR-PKSCET scheme is demonstrated to possess four security properties, namely, leakage resilience, indistinguishability, one-wayness, and existential unforgeability. Based on the proposed LR-PKSCET scheme, an anti-scam system (application) is presented to mitigate the ongoing occurrence of a myriad of scam cases.

Key words: leakage-resilient, side-channel attacks, equality test, signcryption.

1. Introduction

In recent years, as data has significantly grown, there is an increasing inclination among people to store data in cloud servers. Cloud computing (Sun, 2020; Alouffi *et al.*, 2021), viewed as an efficient way of data storage and processing, is gradually becoming an indispensable infrastructure in various industry applications. However, data transmitting and storing in the cloud have also raised security concerns about data confidentiality. Therefore, to enhance the demand for data privacy protection is a significant issue. To ensure the security of data transmission in the cloud, encryption technology has become an essential topic in order to prevent unauthorized individuals from accessing sensitive information. Public key encryption (Lee *et al.*, 2019; Deverajan *et al.*, 2021; Zhou *et al.*, 2021) is a well-known technique employed to ensure data confidentiality.

Since data in a cloud environment is encrypted for confidentiality, it becomes a challenge to find a specific data in cloud effectively. To address this issue, the concept of public key encryption with keyword search (PKEKS) (Boneh *et al.*, 2004) has been proposed to achieve an effective search functionality for encrypted data. However, a PKEKS scheme has a limitation in the sense that it can only search for ciphertexts encrypted under identical receiver (entity) with the same public key. To overcome this limitation, the concept of

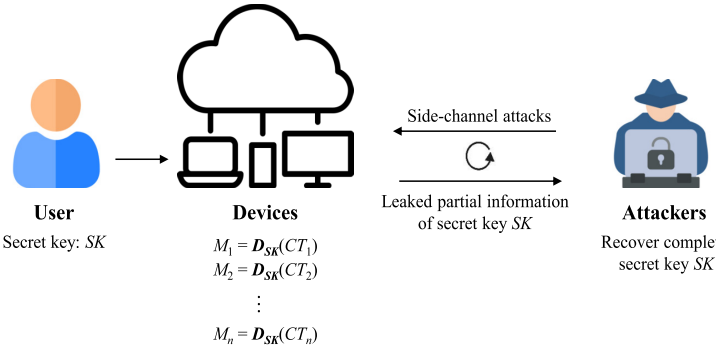

Fig. 1. Recover complete secret key *SK* by side-channel attacks.

public key encryption with equality test (PKEET) (Yang *et al.*, 2010) has been introduced. A PKEET scheme allows users to perform comparative searches on ciphertexts encrypted under different public keys without revealing sensitive data.

Recently, a new type of attack has been actively discussed, known as side-channel attacks (Kubota *et al.*, 2021; Ngo *et al.*, 2021). A side-channel attack refers to a situation that attackers do not attempt to directly break the cryptographic algorithm in a security system, but exploit side-channel information, such as power consumption or timing, to gain access to critical system information. For example, as shown in Fig. 1, a user employs her/his secret key *SK* to perform the *decryption* algorithm by inputting the ciphertext *CT* at the device. Upon the completion of the decryption, the plaintext *M* is obtained. During the decryption process, an attacker could launch side-channel attacks to obtain crucial partial information about the user's secret key *SK*. By doing so, the attacker can acquire partial information of the user's secret key in every decryption process. After several rounds, the attacker may potentially reconstruct the complete sensitive data or the user's secret key *SK* by analysing these side-channel signals.

It is worth noting that a cryptographic mechanism known as leakage-resilient public key encryption (LR-PKE) (Akavia *et al.*, 2009) has indeed been studied and published. It can withstand such side-channel attacks mentioned above. However, this mechanism's application in diverse cloud computing environments remains constrained due to the absence of the property of equality test of ciphertext. Furthermore, to ensure the authenticity and integrity of information is also an important issue. Simultaneously, the development of cryptographic mechanisms that can efficiently merge digital signature and encryption has emerged as a pivotal topic in this field. In light of these challenges, this paper endeavours to propose a novel scheme, called leakage-resilient public key signcryption with equality test (LR-PKSCET), which not only offers robust security against side-channel attacks but also seamlessly integrates the processes of digital signing and encryption. In light of the increasing incidence of fraudulent activities, we will leverage the proposed LR-PKSCET as the foundation for developing an anti-scam system application. Our specific contributions include the following.

– The framework and security notions of LR-PKSCET: We establish the framework of LR-PKSCET that includes six distinct algorithms, and present the associated security

notions of LR-PKSCET. The security notions encompass leakage resilience, indistinguishability, one-wayness, and existential unforgeability.
– A concrete LR-PKSCET scheme: Under the framework of LR-PKSCET, we propose a concrete LR-PKSCET scheme that meets the defined security notions.
– Security analysis: By using hash function properties and discrete logarithm problem, we provide a rigorous security proof of the proposed scheme under the associated security notions.
– Comparison with other schemes: Compared to existing schemes, our proposed LR-PKSCET scheme distinguishes itself as the first to withstand side-channel attacks.
– LR-PKSCET's application: We extend the applicability of the proposed scheme to anti-scam systems that aims to mitigate the ongoing occurrence of a myriad of scam cases.

The remaining sections of this paper include the following parts. Section 2 introduces related work. Preliminaries are given in Section 3. Section 4 shows the LR-PKSCET framework and its associated security definitions. The concrete LR-PKSCET scheme is presented in Section 5. Formal proofs of ensuring the security of the LR-PKSCET scheme are given in Section 6. Performance analysis is carried out in Section 7. Section 8 introduces an application. Lastly, Section 9 offers the concluding remarks.

2. Related Work

The concept of the PKEET scheme was first introduced by Yang *et al.* (2010) as an extension of the existing public key encryption with keyword search (Boneh *et al.*, 2004). The primary goal of their research was to enable the comparison of encrypted data, or ciphertexts, generated under different public keys. Based on the Yang *et al.*'s work (2010), a significant amount of related research on PKEET has been continuously conducted and published. Tang (2011) designed a PKEET scheme with a proxy, where the proxy can obtain tokens from different users and use them to perform equality test of associated ciphertexts. Based on this authorization concept, Tang (2012a) introduced a PKEET scheme that incorporated user-specified authorization. Subsequently, Tang (2012b) further extended the PKEET scheme (Tang, 2012a) to offer the support for fine-grained authorization. Huang *et al.* (2014) introduced a novel PKEET scheme which enables the comparison of ciphertexts without requiring decryption. As a result, their scheme achieves the verification of equivalence among ciphertexts encrypted using the PKEET scheme. Building on this foundation, Huang *et al.* (2015) expanded Huang *et al.*'s scheme (2014) to propose a public key encryption with authorized equality test that allowed authorized proxy to perform equality test on selected ciphertexts. This enhancement augmented the versatility and functionality of the PKEET scheme. Another significant advancement in this field was made by Ma *et al.* (2015). They introduced a PKEET scheme with four distinct authorization policies, thereby increasing the adaptability of PKEET. However, these proposed PKEET schemes were all shown to be secure only under the random oracle model. To overcome this limitation, Lee *et al.* (2020) presented a generic construction of a PKEET scheme in the standard model which provides enhanced security guarantees. Additionally, for the lattice-based

cryptography, Duong *et al.* (2019) introduced a PKEET scheme in the standard model which was built on lattice concepts from an identity-based encryption scheme (Agrawal *et al.*, 2010). On the other hand, to simultaneously ensure the confidentiality, authenticity, and integrity of messages, Le *et al.* (2021) proposed a lattice-based signcryption with equality test in the standard model.

In the real-world scenarios of public key systems, the significance of secret keys is widely acknowledged. However, it is worth noting that the secret keys are often stored in the devices, making them susceptible to potential side-channel attacks (Kubota *et al.*, 2021; Ngo *et al.*, 2021). To avoid the scenario that secret keys can be computed when facing such attacks, the leakage-resilient cryptography (Dziembowski and Pietrzak, 2008; Faust *et al.*, 2010) has emerged as a crucial topic. The core ambition of leakage-resilient cryptography was the formulation of algorithms with the capability to effectively counteract side-channel attacks. For data confidentiality, Akavia *et al.* (2009) introduced a leakage-resilient public key encryption (LR-PKE) scheme that effectively safeguards sensitive information even in the presence of potential side-channel threats. Subsequently, several studies (Naor and Segev, 2009, 2012; Li *et al.*, 2013) related to LR-PKE have also been published to enhance both security and efficiency. However, the aforementioned LR-PKE schemes are specifically designed to provide security only in a bounded leakage model. To overcome this limitation, Kiltz and Pietrzak (2010) proposed the first LR-PKE scheme to provide security in continuous leakage models. Furthermore, Galindo *et al.* (2016) also presented a LR-PKE that drew inspiration from ElGamal and offered security in the continuous leakage model. For addressing the issue of ensuring the security of public key encryption when the randomness of ciphertexts becomes non-uniform due to faulty implementations or adversarial actions, Huang *et al.* (2022) introduced the concept of leakage-resilient hedged public-key encryption that offered a heightened level of comprehensive security. On the other hand, Tseng *et al.* (2022) introduced a leakage-resilient signcryption to achieve the objectives of message confidentiality, authenticity, and integrity.

3. Preliminaries

3.1. *Bilinear Map*

This section outlines the properties of a bilinear map, which serves as the fundamental basis for the proposed LR-PKSCET scheme discussed in this paper. We choose two multiplicative cyclic groups, denoted by G and G_T, both of a prime order q. Let g be a generator of G. A bilinear map $\hat{e}$: $G \times G \rightarrow G_T$ satisfies the following properties.

(1) Bilinear property: For any $a, b \in Z_q^*$, $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$.
(2) Non-degenerate property: $\hat{e}(g, g) \neq 1$.
(3) Computable property: The computation of $\hat{e}(A, B)$ is efficient for any given $A, B \in G$.

3.2. *Generic Bilinear Group Model*

The generic bilinear group (GBG) model, introduced by Boneh *et al.* (2005), is served as a security analysis technique for cryptographic schemes. This model is used in the security game of a cryptographic scheme where an adversary and a challenger interact with each other. Initially, the challenger creates the bilinear parameters $G, G_T, q, g, \hat{e}$ defined above. During the adversary's operations in the bilinear parameters, the adversary can request three types of queries from the challenger. The three types of queries consist of the multiplicative query Q_G of G, the multiplicative query Q_{G_T} of G_T, and the bilinear pairing query $Q_{\hat{e}}$ of $\hat{e}$. Additionally, the challenger establishes two injective random mappings, $F_1 : Z_q^* \to \mathbb{B}G$ and $F_2 : Z_q^* \to \mathbb{B}G_T$, to encode all the elements of G and G_T into distinct bit strings. These mappings must satisfy the conditions $\mathbb{B}G \cap \mathbb{B}G_T = \phi$ and $|\mathbb{B}G| = |\mathbb{B}G_T| = q$. The behaviours of the three associated queries Q_G, Q_{G_T}, and $Q_{\hat{e}}$, for m and n in Z_q^*, are defined as follows:

- $Q_G(F_1(m), F_1(n)) \to F_1(m + n \bmod q)$.
- $Q_{G_T}(F_2(m), F_2(n)) \to F_2(m + n \bmod q)$.
- $Q_{\hat{e}}(F_1(m), F_1(n)) \to F_1(m \cdot n \bmod q)$.

3.3. *Security Assumptions and Entropy*

To establish the security of the LR-PKSCET scheme, we depend on the computational complexity of the discrete logarithm problem (DL) and the properties of hash functions (HF). More precisely, our security analysis is based on the following two related assumptions.

- DL assumption: The DL problem aims to find the unknown value $x \in Z_q^*$ from a given g^x. If there exists a probabilistic polynomial-time (PPT) adversary, the probability of this adversary successfully determining $x \in Z_q^*$ is considered negligible.
- HF assumption: Given a hash function HF, which possesses collision-resistant and one-way properties, the probability of a PPT adversary successfully finding two values Y_1, Y_2. such that $HF(Y_1) = HF(Y_2)$, is considered negligible.

To assess the probability of successfully obtaining a secret key through side-channel attacks, we utilize entropy concept to measure the probability of successful reconstruction. In the following, we introduce two types of min-entropies for conducting this analysis.

- The min-entropy of a finite random variable M is denoted by $H_\infty(M) = -\log_2(\max_m \Pr[M = m])$.
- The average conditional min-entropy of a finite random variable M under a condition X is denoted by $\widetilde{H}_\infty(M|X) = -\log_2(X[\max_m \Pr[M = m|X]])$.

To measure entropy, there are two cases to be considered: one with a single secret key and the other with multiple secret keys. For these two cases, we will refer to the results of Dodis *et al.* (2008) (Lemma 1) and Galindo and Virek (2013) (Lemma 2), respectively.

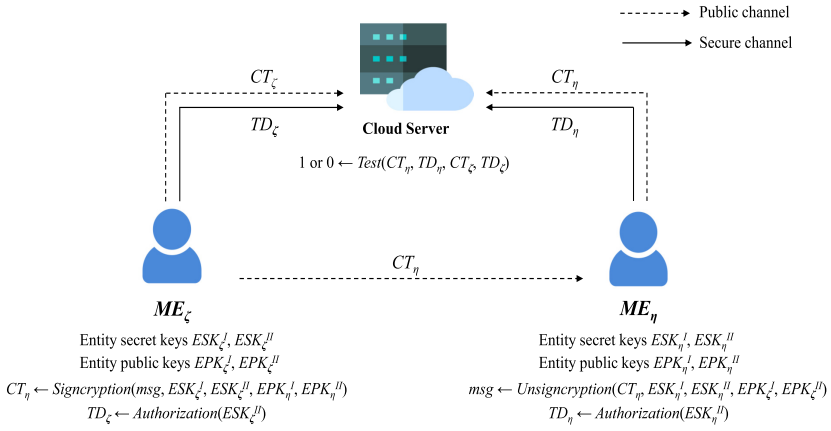

Fig. 2. The framework of LR-PKSCET.

Lemma 1. *Consider a single secret key M and let Φ represent the maximum length of the bit strings that might be leaked from the secret key M. We define $f : M \rightarrow \{0,1\}^\Phi$ as the leakage function. In this context, we obtain $\widetilde{H}_\infty(M|f(M)) \geqq H_\infty(M) - \Phi$.*

Lemma 2. *Consider n multiple secret keys, denoted as $M_1, M_2, \ldots, M_n$. Let $F \in Z_q[M_1, M_2, \ldots, M_n]$ be a polynomial with degree at most d. For each $i = 1, 2, \ldots, n$, let PD_i be a probability distribution of $M_i = m_i$ such that $H_\infty(PD_i) \geqq \log q - \Phi$, where $0 \leqq \Phi \leqq \log q$. If $m_i \overset{PD_i}{\leftarrow} Z_q$ are independent, the inequality $\Pr[F(M_1 = m_1, M_2 = m_2, \ldots, M_n = m_n) = 0] \leqq (d/q)2^\Phi$ holds. If $\Phi < \log q(1-\varepsilon)$, $\Pr[F(M_1 = m_1, M_2 = m_2, \ldots, M_n = m_n) = 0]$ is negligible, where ε is a positive decimal.*

4. Framework and Security Notions for LR-PKSCET

We illustrate the framework of our LR-PKSCET scheme in Fig. 2, which includes a cloud server and two entities (also works more than two member entities). The two member entities ME_ζ and ME_η can respectively generate their own entity secret keys denoted by $(ESK_\zeta^I, ESK_\zeta^{II})$ and $(ESK_\eta^I, ESK_\eta^{II})$ and entity public keys denoted by $(EPK_\zeta^I, EPK_\zeta^{II})$ and $(EPK_\eta^I, EPK_\eta^{II})$. The entity ME_ζ utilizes her/his entity secret keys $(ESK_\zeta^I, ESK_\zeta^{II})$ along with the entity public keys $(EPK_\eta^I, EPK_\eta^{II})$ of the entity ME_η to perform the process of signing and encryption on the message *msg* using the *Signcryption* algorithm. The output ciphertext CT_η will be sent to the entity ME_η. Upon receiving CT_η, the entity ME_η can utilize her/his entity secret keys $(ESK_\eta^I, ESK_\eta^{II})$ along with the entity public keys $(EPK_\zeta^I, EPK_\zeta^{II})$ of the entity ME_ζ to perform a decryption and verification process using the *Unsigncryption* algorithm. The process above allows the entity ME_ζ to obtain the message *msg*. On the other hand, the entities ME_ζ and ME_η can respectively transmit their ciphertexts CT_ζ, CT_η, and trapdoors TD_ζ, TD_η to the cloud server. The cloud server is capable of testing whether the two ciphertexts contain the same plaintext.

Table 1
Symbols.

Symbol	Meaning
DE	The designated entity of the system
λ	The security parameter
SP	The system parameters
ESK^{I}	The first entity secret key
ESK^{II}	The second entity secret key
EPK^{I}	The first entity public key
EPK^{II}	The second entity public key
$ESK^{I}_{S,i-1}$	The first entity secret key of sender in the $i-1$th round
$ESK^{II}_{S,i-1}$	The second entity secret key of sender in the $i-1$th round
$ESK^{I}_{R,j-1}$	The first entity secret key of receiver in the $j-1$th round
$ESK^{II}_{R,j-1}$	The second entity secret key of receiver in the $j-1$th round
ESK^{II}_{k-1}	The second entity secret key in the $k-1$th round
TD	The trapdoor
msg	The message
CT	The ciphertext

4.1. *Framework for LR-PKSCET*

In this subsection, we provide a formal definition of the proposed scheme's framework, which is defined according to the frameworks of PKSCET (Le *et al.*, 2021) and LR-PKSC (Tseng *et al.*, 2022). To facilitate understanding of the formal definition, a symbol table is provided in Table 1.

Definition 1. An LR-PKSCET scheme is composed of six algorithms, namely, *Initialization*, *EntityKeyGen*, *Signcryption*, *Unsigncryption*, *Authorization*, and *Test* as follows.

- *Initialization*: The designated entity DE of this scheme (system) is responsible for executing the algorithm with a security parameter λ and outputs the system parameters SP.
- *EntityKeyGen*: By executing the algorithm with the system parameters SP, the member entity ME generates two entity secret keys ESK^{I}, ESK^{II} and two entity public keys EPK^{I}, EPK^{II}.
- *Signcryption*: By executing the algorithm in the ith round with the system parameters SP, two entity secret keys $ESK^{I}_{S,i-1}$, $ESK^{II}_{S,i-1}$ (the $i-1$th round) of the member entity ME_S, identified as the sender, a message msg, and two entity public keys EPK^{I}_{R}, EPK^{II}_{R} of the member entity ME_R, identified as the receiver, the sender ME_S generates a ciphertext CT.
- *Unsigncryption*: By executing the algorithm in the jth round with the system parameters SP, two entity secret keys $ESK^{I}_{R,j-1}$, $ESK^{II}_{R,j-1}$ (the $j-1$th round) of a member entity ME_R identified as the receiver, a ciphertext CT, and two entity public keys EPK^{I}_{S}, EPK^{II}_{S} of a member entity ME_S identified as the sender, the sender ME_S generates a message msg.

– *Authorization*: By executing the algorithm in the kth round with the system parameters SP and the entity secret key ESK_{k-1}^{II} (the $k-1$th round) of a member entity ME, the member entity ME generates a trapdoor TD.
– *Test*: By executing the algorithm with the system parameters SP, the ciphertext-trapdoor pair (CT_ζ, TD_ζ) of a member entity ME_ζ and the ciphertext-trapdoor pair (CT_η, TD_η) of a member entity ME_η, the cloud server CS generates 1 or 0.

4.2. *Security Notions for LR-PKSCET*

By the technique introduced in Galindo and Virek (2013), we initiate six unique leakage functions: $LF_{SC,i}^{A}$, $LF_{SC,i}^{B}$, $LF_{USC,j}^{A}$, $LF_{USC,j}^{B}$, $LF_{Auth,k}^{A}$ and $LF_{Auth,k}^{B}$. Specifically, $LF_{SC,i}^{A}$ and $LF_{SC,i}^{B}$ are utilized to extract leaked information of ESK_S^I and ESK_S^{II}, denoted as $ESK_S^I = (ESK_{S,i,A}^I, ESK_{S,i,B}^I)$ and $ESK_S^{II} = (ESK_{S,i,A}^{II}, ESK_{S,i,B}^{II})$ which are used in the *Signcryption* algorithm during the ith round. Similarly, $LF_{USC,j}^{A}$ and $LF_{USC,j}^{B}$ are applied to extract leaked information of ESK_R^I and ESK_R^{II}, denoted as $ESK_R^I = (ESK_{R,j,A}^I, ESK_{R,j,B}^I)$ and $ESK_R^{II} = (ESK_{R,j,A}^{II}, ESK_{R,j,B}^{II})$ which are used in the *Unsigncryption* algorithm during the jth round. The last two leakage functions $LF_{Auth,k}^{A}$ and $LF_{Auth,k}^{B}$ are utilized to extract leaked information of ESK_R^{II}, denoted as $ESK_R^{II} = (ESK_{R,k,A}^{II}, ESK_{R,k,B}^{II})$ which is used in the *Authorization* algorithm during the kth round. Let Φ represent the maximum length of the bit strings that might be leaked from the secret keys. As a result, the size of $|LF_{SC,i}^{A}|$, $|LF_{SC,i}^{B}|$, $|LF_{USC,j}^{A}|$, $|LF_{USC,j}^{B}|$, $|LF_{Auth,k}^{A}|$, and $|LF_{Auth,k}^{B}|$ are all bounded by Φ, where the notation $|\cdot|$ signifies the length of a bit string. Also, for convenience we will adopt the following six symbols:

– $\Lambda LF_{SC,i}^{A} = LF_{SC,i}^{A}(ESK_{S,i,A}^I, ESK_{S,i,A}^{II})$;
– $\Lambda LF_{SC,i}^{B} = LF_{SC,i}^{B}(ESK_{S,i,B}^I, ESK_{S,i,B}^{II})$;
– $\Lambda LF_{USC,j}^{A} = LF_{USC,j}^{A}(ESK_{R,j,A}^I, ESK_{R,j,A}^{II})$;
– $\Lambda LF_{USC,j}^{B} = LF_{USC,j}^{B}(ESK_{R,j,B}^I, ESK_{R,j,B}^{II})$;
– $\Lambda LF_{Auth,k}^{A} = LF_{Auth,k}^{A}(ESK_{R,k,A}^{II})$;
– $\Lambda LF_{Auth,k}^{B} = LF_{Auth,k}^{B}(ESK_{R,k,B}^{II})$.

Next, we define indistinguishable security, one-way security, and existential unforgeability to serve as the security notion that can withstand adversaries with the capabilities of side-channel attacks.

Definition 2. If the advantage of an adversary $\mathcal{A}_I$ to break a LR-PKSCET scheme in the following security game under side-channel and chosen-ciphertext attacks is negligible, we say that the scheme has leakage resilience and indistinguishable security.

– *Setup*: The challenger $\mathcal{CH}$ is responsible for executing the *Initialization* algorithm with a security parameter λ and outputs the system parameters SP which will be made public.
– *Phase* 1: The adversary $\mathcal{A}_I$ has the capability to make various adaptive queries as follows.

- ✓ *EntityKeyGen query*: $\mathcal{A}_I$ sends a query containing member entity's information ME. Subsequently, $\mathcal{CH}$ processes the *EntityKeyGen* algorithm to acquire two entity secret keys ESK^I, ESK^{II} and two entity public keys EPK^I, EPK^{II}, and returns them back to $\mathcal{A}_I$.
- ✓ *Signcryption query*: $\mathcal{A}_I$ sends a query containing two entity secret keys $ESK^I_{S,i}$, $ESK^{II}_{S,i}$ of a member entity ME_S identified as the sender, a message msg, and two entity public keys EPK^I_R, EPK^{II}_R of a member entity ME_R identified as the receiver. Subsequently, $\mathcal{CH}$ processes the *Signcryption* algorithm to acquire a ciphertext CT, and returns it back to $\mathcal{A}_I$.
- ✓ *Signcryption leak query*: $\mathcal{A}_I$ sends a query containing an index i, and two leaked functions $LF^A_{SC,i}$ and $LF^B_{SC,i}$. Subsequently, $\mathcal{CH}$ returns two leaked information $\Lambda LF^A_{SC,i}$ and $\Lambda LF^B_{SC,i}$ back to $\mathcal{A}_I$.
- ✓ *Unsigncryption query*: $\mathcal{A}_I$ sends a query containing two entity secret keys $ESK^I_{R,j}$, $ESK^{II}_{R,j}$ of a member entity ME_R identified as the receiver, a ciphertext CT, and two entity public keys EPK^I_S, EPK^{II}_S of a member entity ME_R identified as the sender. Subsequently, $\mathcal{CH}$ processes the *Unsigncryption* algorithm to acquire a message msg, and returns it back to $\mathcal{A}_I$.
- ✓ *Unsigncryption leak query*: $\mathcal{A}_I$ sends a query containing an index j, and two leaked functions $LF^A_{USC,j}$ and $LF^B_{USC,j}$. Subsequently, $\mathcal{CH}$ returns two sets of leaked information $\Lambda LF^A_{USC,j}$ and $\Lambda LF^B_{USC,j}$ back to $\mathcal{A}_I$.
- ✓ *Authorization query*: $\mathcal{A}_I$ sends a query containing the entity secret key ESK^{II}_k of a member entity ME. Subsequently, $\mathcal{CH}$ processes the *Authorization* algorithm to acquire a trapdoor TD, and returns it back to $\mathcal{A}_I$.
- ✓ *Authorization leak query*: $\mathcal{A}_I$ sends a query containing an index k, and two leaked functions $LF^A_{Auth,k}$ and $LF^B_{Auth,k}$. Subsequently, $\mathcal{CH}$ returns two sets of leaked information $\Lambda LF^A_{Auth,k}$ and $\Lambda LF^B_{Auth,k}$ back to $\mathcal{A}_I$.

– *Challenge*: $\mathcal{A}_I$ chooses a specific member entity ME^*_R, and provides two target plaintexts, msg_0 and msg_1, to $\mathcal{CH}$. Subsequently, $\mathcal{CH}$ chooses a random bit $b \in \{0, 1\}$ and utilizes the *Signcryption* algorithm in the ith round with the corresponding parameters msg^*_b, EPK^I_R, EPK^{II}_R, $ESK^I_{S,i}$ and $ESK^{II}_{S,i}$ to generate the target ciphertext CT^*. Then, $\mathcal{CH}$ sends ciphertext CT^* to $\mathcal{A}_I$. One restriction is that ME^*_R is not allowed to appear in the *EntityKeyGen queries*.
– *Phase* 2: The adversary $\mathcal{A}_I$ can make further queries as in the *Phase* 1 except that the selected target, namely ME^*_R, msg^*_b, $ESK^I_{S,i}$ and $ESK^{II}_{S,i}$ are not allowed to appear in the *EntityKeyGen*, the *Unsigncryption* and the *Authorization queries*.
– *Guess phase*: $\mathcal{A}_I$ produces the value b' and succeeds in the game if b' is equal to b. The advantage of winning the game is represented by $Adv(\mathcal{A}_I) = |\Pr[b' = b] - 1/2|$.

Definition 3. If the advantage of an adversary $\mathcal{A}_{II}$ to break a LR-PKSCET scheme in the following game under side-channel and chosen-ciphertext attacks is negligible, we say that the scheme has leakage resilience and one-way security.

– *Setup*: This stage is the same as that in Definition 2.

- *Phase* 1: This stage is the same as that in Definition 2.
- *Challenge*: $\mathcal{A}_{II}$ chooses a specific member entity ME_R^* to $\mathcal{CH}$. Subsequently, $\mathcal{CH}$ chooses a random message msg^* and utilizes the *Signcryption* algorithm with msg^*, EPK_R^I, EPK_R^{II}, $ESK_{S,i}^I$ and $ESK_{S,i}^{II}$ to generate the target ciphertext CT^*. Then, $\mathcal{CH}$ sends ciphertext CT^* to $\mathcal{A}_{II}$. One restriction is that ME_R^* is not allowed to appear in the *EntityKeyGen* and the *Authorization queries*.
- *Phase* 2: The adversary $\mathcal{A}_{II}$ can make further queries as in the *Phase* 1 except that the selected target, namely ME_R^*, msg^*, $ESK_{S,i}^I$ and $ESK_{S,i}^{II}$ are not allowed to appear in the *EntityKeyGen* and the *Unsigncryption queries*.
- *Guess phase*: $\mathcal{A}_{II}$ produces the message msg' and succeeds in the game if msg' is equal to msg^*. The advantage of winning the game is represented by $Adv(\mathcal{A}_{II}) = |\Pr[msg' = msg^*]|$.

Definition 4. If the advantage of an adversary $\mathcal{A}_{III}$ to break a LR-PKSCET scheme in the following game under side-channel and chosen-message attacks is negligible, we say that the scheme has leakage resilience and existential unforgeability.

- *Setup*: This stage is the same as that in Definition 2.
- *Phase* 1: This stage is the same as that in Definition 2.
- *Forgery*: The adversary $\mathcal{A}_{III}$ successfully forges a ciphertext $CT^* = (ME_S^*, ME_R^*, U^*, V^*, R^*, S^*, \sigma^*)$ for a message M^*, and we declare $\mathcal{A}_{III}$ as the winner of this game if the following conditions are satisfied.
  - ✓ The *Unsigncryption* algorithm is capable of generating the message M^*.
  - ✓ The *Signcryption queries* are not allowed to query the message M^*, ME_S^* or ME_R^*.
  - ✓ The *EntityKeyGen queries* are not allowed to query the member entity ME_S^*.

5. The Proposed LR-PKSCET Scheme

In this section, we show a leakage-resilient public key signcryption with equality test (LR-PKSCET) scheme that includes the following six algorithms.

- *Initialization*: The designated entity *DE* of this scheme (system) is responsible for executing the algorithm with a security parameter λ and outputs the system parameters *SP* by the following steps.
  - ✓ Follow the guidelines provided in Section 3 to generate the bilinear parameters G, G_T, q, $\hat{e}$, g.
  - ✓ Pick two random values $x, y \in Z_q^*$, and then compute $X = g^x$ and $Y = g^y$.
  - ✓ Select five hash functions, namely, $HF_1 : G_T \rightarrow G$, $HF_2 : G \times G \times G_T \rightarrow \{0,1\}^{m+n}$, $HF_3 : \{0,1\}^m \rightarrow G$, $HF_4 : \{0,1\}^{m+n} \rightarrow Z_q^*$, $HF_5 : \{0,1\}^* \times \{0,1\}^{m+n} \times \{0,1\}^m \times G \times G \times G \rightarrow \{0,1\}^t$, where m, n and t represent fixed lengths.
  - ✓ Define the system parameters $SP = \{G, G_T, q, \hat{e}, g, X, Y, HF_1, HF_2, HF_3, HF_4, HF_5\}$.
- *EntityKeyGen*: By executing the algorithm with the system parameters *SP*, the member entity *ME* generates two entity secret keys ESK^I, ESK^{II} and two entity public keys EPK^I, EPK^{II} using the following steps.

✓ Choose two random values $\alpha, \beta \in Z_q^*$, and then compute $ESK^I = g^\alpha$ and $ESK^{II} = g^\beta$.

✓ Utilize ESK^I and ESK^{II} to establish $EPK^I = \hat{e}(g, g^\alpha)$ and $EPK^{II} = \hat{e}(g, g^\beta)$, respectively.

Next, the member entity ME chooses two random renewed values, a and b, to calculate the initial entity secret keys $ESK_0^I = (ESK_{0,A}^I, ESK_{0,B}^I) = (ESK^I \cdot g^a, g^{-a})$ and $ESK_0^{II} = (ESK_{0,A}^{II}, ESK_{0,B}^{II}) = (ESK^{II} \cdot g^b, g^{-b})$.

– *Signcryption*: By executing the algorithm in the ith round with the system parameters SP, two entity secret keys $ESK_{S,i-1}^I$, $ESK_{S,i-1}^{II}$ (the $i-1$th round) of the member entity ME_S, identified as the sender, a message msg, and two entity public keys EPK_R^I, EPK_R^{II} of the member entity ME_R, identified as the receiver, the sender ME_S generates a ciphertext CT by the following steps.

✓ Pick two random values $h \in \{0,1\}^n$ and $v \in Z_q^*$, and compute $U = g^u$ and $V = g^v$, where $u = HF_4(msg, h)$.

✓ Compute $R = HF_2((EPK_R^I)^v, U, V) \oplus (msg||h)$ and $S = HF_1((EPK_R^{II})^v) \cdot HF_3(msg)^u$.

✓ Choose a random values $c \in Z_q^*$ which are used to update the entity secret keys.

✓ Compute two entity secret keys $ESK_{S,i}^I = (ESK_{S,i,A}^I, ESK_{S,i,B}^I) = (ESK_{S,i-1,A}^I \cdot g^c, ESK_{S,i-1,B}^I \cdot g^{-c})$ and $ESK_{S,i}^{II} = (ESK_{S,i,A}^{II}, ESK_{S,i,B}^{II}) = (ESK_{S,i-1,A}^{II} \cdot g^c, ESK_{S,i-1,B}^{II} \cdot g^{-c})$.

✓ Compute $\delta = HF_5(ME_S, ME_R, U, V, R, S, msg)$, and then set $TS^I = ESK_{S,i,A}^I \cdot (X \cdot Y^\delta)^u$ and $TS^{II} = ESK_{S,i,A}^{II} \cdot (X \cdot Y^\delta)^v$.

✓ Generate a signature $\sigma = ESK_{S,i,B}^I \cdot TS^I \cdot ESK_{S,i,B}^{II} \cdot TS^{II}$.

✓ Set a ciphertext $CT = (ME_S, ME_R, U, V, R, S, \sigma)$.

– *Unsigncryption*: By executing the algorithm in the jth round with the system parameters SP, two entity secret keys $ESK_{R,j-1}^I$, $ESK_{R,j-1}^{II}$ (the $j-1$th round) of the member entity ME_R, identified as the receiver, a ciphertext CT, and two entity public keys EPK_S^I, EPK_S^{II} of the member entity ME_S, identified as the sender, the sender ME_S generates a message msg' by the following steps.

✓ Choose a random values $d \in Z_q^*$ which are used to update the entity secret keys.

✓ Compute two entity secret keys $ESK_{R,j}^I = (ESK_{R,j,A}^I, ESK_{R,j,B}^I) = (ESK_{R,j-1,A}^I \cdot g^d, ESK_{R,j-1,B}^I \cdot g^{-d})$ and $ESK_{R,j}^{II} = (ESK_{R,j,A}^{II}, ESK_{R,j,B}^{II}) = (ESK_{R,j-1,A}^{II} \cdot g^d, ESK_{R,j-1,B}^{II} \cdot g^{-d})$.

✓ Set $TU^I = \hat{e}(ESK_{R,j,A}^I, V)$ and $TU^{II} = \hat{e}(ESK_{R,j,A}^{II}, V)$.

✓ Compute $R \oplus HF_2(TU^I \cdot \hat{e}(ESK_{R,j,B}^I, V), U, V)$ to obtain msg' and h'.

✓ Set $u' = HF_4(msg', h')$. If the two equations $U = g^{u'}$ and $S = HF_1(TU^{II} \cdot \hat{e}(ESK_{R,j,B}^{II}, V)) \cdot HF_3(msg')^{u'}$ hold, compute $\delta' = HF_5(ME_S, ME_R, U, V, R, S, msg')$. Otherwise, return the result "failure".

✓ If $\hat{e}(g, \sigma) = EPK^I \cdot EPK^{II} \cdot \hat{e}(U \cdot V, X \cdot Y^{\delta'})$ holds, return the message msg'. Otherwise, return the result "failure".

– *Authorization*: By executing the algorithm in the kth round with the system parameters SP and the entity secret key $ESK_{S,k-1}^{II}$ (the $k-1$th round) of a member entity ME, the

member entity *ME* generates a trapdoor *TD* using the following steps.

✓ Choose a random value $e \in Z_q^*$, and update the entity secret key $ESK_k^{II} = (ESK_{k,A}^{II}, ESK_{k,B}^{II}) = (ESK_{k-1,A}^{II} \cdot g^e, ESK_{k-1,B}^{II} \cdot g^{-e})$.

✓ Set $TT = ESK_{k,A}^{II}$, and compute $TD = TT \cdot ESK_{k,B}^{II}$.

– *Test*: By executing the algorithm with the system parameters *SP*, the ciphertext-trapdoor pair (CT_ζ, TD_ζ) of the member entity ME_ζ and the ciphertext-trapdoor pair (CT_η, TD_η) of the member entity ME_η, the cloud server *CS* generates 1 or 0 by the following steps.

✓ Compute $R_\zeta = S_\zeta / HF_1(\hat{e}(TD_\zeta, V_\zeta)$ and $R_\eta = S_\eta / HF_1(\hat{e}(TD_\eta, V_\eta)$.

✓ If the equation $\hat{e}(R_\eta, U_\zeta) = \hat{e}(R_\zeta, U_\eta)$ holds, output 1. Otherwise, output 0.

The following describes how to obtain the message *msg* and verify the signature σ in the *Unsigncryption* algorithm.

$$
\begin{aligned}
&R \oplus HF_2\big(TU^I \cdot \hat{e}\big(ESK_{R,j,B}^I, V\big), U, V\big) \\
&\quad = HF_2\big(\big(EPK_R^I\big)^v, U, V\big) \oplus (msg \,\|\, h) \\
&\qquad \oplus HF_2\big(\hat{e}\big(ESK_{R,j,A}^I, V\big) \cdot \hat{e}\big(ESK_{R,j,B}^I, V\big), U, V\big) \\
&\quad = (msg \,\|\, h) \oplus HF_2\big(\big(EPK_R^I\big)^v, U, V\big) \oplus HF_2\big(\hat{e}\big(ESK_R^I, V\big), U, V\big) \\
&\quad = (msg \,\|\, h) \oplus HF_2\big(\hat{e}\big(g, g^\alpha\big)^v, U, V\big) \oplus HF_2\big(\hat{e}\big(g^\alpha, g^v\big), U, V\big) \\
&\quad = (msg \,\|\, h) \oplus HF_2\big(\hat{e}(g, g)^{\alpha v}, U, V\big) \oplus HF_2\big(\hat{e}(g, g)^{\alpha v}, U, V\big) \\
&\quad = (msg \,\|\, h), \\
&\hat{e}(g, \sigma) = \hat{e}\big(g, ESK_{S,i,B}^I \cdot TS^I \cdot ESK_{S,i,B}^{II} \cdot TS^{II}\big) \\
&\qquad = \hat{e}\big(g, ESK_{S,i,B}^I \cdot ESK_{S,i,A}^I \cdot \big(X \cdot Y^\delta\big)^u \cdot ESK_{S,i,B}^{II} \cdot ESK_{S,i,A}^{II} \cdot \big(X \cdot Y^\delta\big)^v\big) \\
&\qquad = \hat{e}\big(g, ESK_S^I \cdot \big(X \cdot Y^\delta\big)^u \cdot ESK_S^{II} \cdot \big(X \cdot Y^\delta\big)^v\big) \\
&\qquad = \hat{e}\big(g, g^\alpha \cdot \big(X \cdot Y^\delta\big)^u \cdot g^\beta \cdot \big(X \cdot Y^\delta\big)^v\big) \\
&\qquad = \hat{e}\big(g, g^\alpha\big) \cdot \hat{e}\big(g, g^\beta\big) \cdot \hat{e}\big(g, \big(X \cdot Y^\delta\big)^u\big) \cdot \hat{e}\big(g, \big(X \cdot Y^\delta\big)^v\big) \\
&\qquad = EPK^I \cdot EPK^{II} \cdot \hat{e}\big(g^u, \big(X \cdot Y^\delta\big)\big) \cdot \hat{e}\big(g^v, \big(X \cdot Y^\delta\big)\big) \\
&\qquad = EPK^I \cdot EPK^{II} \cdot \hat{e}\big(U, \big(X \cdot Y^\delta\big)\big) \cdot \hat{e}\big(V, \big(X \cdot Y^\delta\big)\big) \\
&\qquad = EPK^I \cdot EPK^{II} \cdot \hat{e}\big(U \cdot V, X \cdot Y^\delta\big).
\end{aligned}
$$

Next, we present the equation derivation process used in the $Test$ algorithm.

$$
\begin{aligned}
R_\zeta &= S_\zeta / HF_1\big(\hat{e}(TD_\zeta, V_\zeta)\big) \\
&= HF_1\big(\big(EPK_\zeta^{II}\big)^{v_\zeta}\big) \cdot HF_3(msg_\zeta)^{u_\zeta} / HF_1\big(\hat{e}\big(TT_\zeta \cdot ESK_{k,B}^{II}, g^{v_\zeta}\big)\big) \\
&= HF_1\big(\hat{e}\big(g, g^{\beta_\zeta}\big)^{v_\zeta}\big) \cdot HF_3(msg_\zeta)^{u_\zeta} / HF_1\big(\hat{e}\big(ESK_{\zeta,k,A}^{II} \cdot ESK_{\zeta,k,B}^{II}, g^{v_\zeta}\big)\big) \\
&= HF_1\big(\hat{e}\big(g^{v_\zeta}, g^{\beta_\zeta}\big)\big) \cdot HF_3(msg_\zeta)^{u_\zeta} / HF_1\big(\hat{e}\big(ESK_\zeta^{II}, g^{v_\zeta}\big)\big) \\
&= HF_1\big(\hat{e}\big(g^{v_\zeta}, g^{\beta_\zeta}\big)\big) \cdot HF_3(msg_\zeta)^{u_\zeta} / HF_1\big(\hat{e}\big(g^{\beta_\zeta}, g^{v_\zeta}\big)\big) \\
&= HF_3(msg_\zeta)^{u_\zeta},
\end{aligned}
$$

$$
\begin{aligned}
R_\eta &= S_\eta / HF_1\big(\hat{e}(TD_\eta, V_\eta)\big) \\
&= HF_1\Big(\big(EPK_\eta^{II}\big)^{v_\eta}\Big) \cdot HF_3(msg_\eta)^{u_\eta} / HF_1\Big(\hat{e}\big(TT_\eta \cdot ESK_{\eta,k,B}^{II}, g^{v_\eta}\big)\Big) \\
&= HF_1\Big(\hat{e}\big(g, g^{\beta_\eta}\big)^{v_\eta}\Big) \cdot HF_3(msg_\eta)^{u_\eta} / HF_1\Big(\hat{e}\big(ESK_{\eta,k,A}^{II} \cdot ESK_{\eta,k,B}^{II}, g^{v_\eta}\big)\Big) \\
&= HF_1\Big(\hat{e}\big(g^{v_\eta}, g^{\beta_\eta}\big)\Big) \cdot HF_3(msg_\eta)^{u_\eta} / HF_1\Big(\hat{e}\big(ESK_\eta^{II}, g^{v_\eta}\big)\Big) \\
&= HF_1\Big(\hat{e}\big(g^{v_\eta}, g^{\beta_\eta}\big)\Big) \cdot HF_3(msg_\eta)^{u_\eta} / HF_1\Big(\hat{e}\big(g^{\beta_\eta}, g^{v_\eta}\big)\Big) \\
&= HF_3(msg_\eta)^{u_\eta}, \\
\hat{e}(R_\eta, U_\zeta) &= \hat{e}\big(HF_3(msg_\eta)^{u_\eta}, g^{u_\zeta}\big) = \hat{e}\big(HF_3(msg_\eta), g\big)^{u_\eta \cdot u_\zeta}, \\
\hat{e}(R_\zeta, U_\eta) &= \hat{e}\big(HF_3(msg_\zeta)^{u_\zeta}, g^{u_\eta}\big) = \hat{e}\big(HF_3(msg_\zeta), g\big)^{u_\zeta \cdot u_\eta}.
\end{aligned}
$$

6. Security Analysis

In this section, we will prove three theorems to establish the security of the proposed LR-PKSCET scheme in terms of indistinguishable security, one-way security and existential unforgeability, while ensuring that the proposed scheme possesses leakage resilience to withstand side-channel attacks.

Theorem 1. *Under the assumptions of DL and HF, the proposed LR-PKSCET scheme possesses leakage resilience and indistinguishable security in the security game* (*Definition* 2) *using the GBG model.*

Proof. Let's begin the security game with the interaction between a challenger $\mathcal{CH}$ and an adversary $\mathcal{A}_I$.

- *Setup*: The challenger $\mathcal{CH}$ utilizes a security parameter λ to execute the *Initialization* algorithm to generate the system parameters $SP = \{G, G_T, q, \hat{e}, g, X, Y, HF_1, HF_2, HF_3, HF_4, HF_5\}$. Furthermore, $\mathcal{CH}$ creates eight lists $\mathcal{L}G$, $\mathcal{L}G_T$, $\mathcal{L}HF_1$, $\mathcal{L}HF_2$, $\mathcal{L}HF_3$, $\mathcal{L}HF_4$, $\mathcal{L}HF_5$, $\mathcal{L}MEkeys$ as below.
  - ✓ The list $\mathcal{L}G$ contains pairs of $(\mathbb{P}G_{r,t,v}, \mathbb{B}G_{r,t,v})$, where each element in G is represented by a multivariate polynomial $\mathbb{P}G_{r,t,v}$, and its corresponding encoded bit-string is denoted by $\mathbb{B}G_{r,t,v}$. Here, r, t, and v respectively represent the query type, the t-th query, and the v-th element in G. Initially, the challenger $\mathcal{CH}$ adds three pairs $(\mathbb{P}g, \mathbb{B}G_{s,0,1})$, $(\mathbb{P}X, \mathbb{B}G_{s,0,2})$ and $(\mathbb{P}Y, \mathbb{B}G_{s,0,3})$ to the list $\mathcal{L}G$.
  - ✓ The list $\mathcal{L}G_T$ contains pairs of $(\mathbb{P}G_{T,r,t,v}, \mathbb{B}G_{T,r,t,v})$, where each element in G_T is represented by a multivariate polynomial $\mathbb{P}G_{T,r,t,v}$, and its corresponding encoded bit-string is denoted by $\mathbb{B}G_{T,r,t,v}$. Furthermore, the symbols r, t, and v have the same meanings as those in the list $\mathcal{L}G$ above.

    Each element present in the lists $\mathcal{L}G$ and $\mathcal{L}G_T$ is represented both as a multivariate polynomial and a bit-string. To facilitate the conversion between these representations, we introduce two conversion rules, namely Rule-1 and Rule-2. These rules illustrate the process of transforming a multivariate polynomial into its corresponding bit-string and vice versa.

- Under Rule-1, when encountering the multivariate polynomial $\mathbb{P}G_{r,t,v}/\mathbb{P}G_{T,r,t,v}$ in the list $\mathcal{L}G/\mathcal{L}G_T$, the procedure involves converting it to the corresponding bit-string $\mathbb{B}G_{r,t,v}/\mathbb{B}G_{T,r,t,v}$, which will be the output. However, if the multivariate polynomial is not present in the list, a random bit-string $\mathbb{B}G_{r,t,v}/\mathbb{B}G_{T,r,t,v}$ related to $\mathbb{P}G_{r,t,v}/\mathbb{P}G_{T,r,t,v}$ is chosen. This newly selected string will be appended to the list $\mathcal{L}G/\mathcal{L}G_T$, and then returned as the output.
- Under Rule-2, when encountering the bit-string $\mathbb{B}G_{r,t,v}/\mathbb{B}G_{T,r,t,v}$ in the list $\mathcal{L}G/\mathcal{L}G_T$, the procedure involves converting it to the corresponding multivariate polynomial $\mathbb{P}G_{r,t,v}/\mathbb{P}G_{T,r,t,v}$, which will be the output. However, if the bit-string is not present in the list, a random polynomial $\mathbb{P}G_{r,t,v}/\mathbb{P}G_{T,r,t,v}$ related to $\mathbb{B}G_{r,t,v}/\mathbb{B}G_{T,r,t,v}$ is chosen. This newly selected polynomial will be appended to the list $\mathcal{L}G/\mathcal{L}G_T$, and then returned as the output.

✓ The list $\mathcal{L}HF_1$ contains pairs of $(\mathbb{P}EPK^{II} \cdot \mathbb{P}V, \mathbb{P}H_1)$, where $\mathbb{P}EPK^{II} \cdot \mathbb{P}V$ and $\mathbb{P}H_1$ are the necessary information for the execution of HF_1.

✓ The list $\mathcal{L}HF_2$ contains pairs of $(\mathbb{P}EPK^{I} \cdot \mathbb{P}V||\mathbb{P}U||\mathbb{P}V, \mathbb{P}H_2)$, where $\mathbb{P}EPK^{I} \cdot \mathbb{P}V||\mathbb{P}U||\mathbb{P}V$ and $\mathbb{P}H_2$ are the necessary information for the execution of HF_2.

✓ The list $\mathcal{L}HF_3$ contains pairs of $(msg, \mathbb{P}H_3)$, where msg and $\mathbb{P}H_3$ are the necessary information for the execution of HF_3.

✓ The list $\mathcal{L}HF_4$ contains pairs of $(msg||h, \mathbb{P}H_4)$, where $msg||h$ and $\mathbb{P}H_4$ are the necessary information for the execution of HF_4.

✓ The list $\mathcal{L}HF_5$ contains pairs of $(ME_S||ME_R||\mathbb{P}U||\mathbb{P}V||\mathbb{P}R||\mathbb{P}S||msg, \mathbb{P}H_5)$, where $ME_S||ME_R||U||\mathbb{P}V||\mathbb{P}R||\mathbb{P}S||msg$ and $\mathbb{P}H_5$ are the necessary information for the execution of HF_5.

✓ The list $\mathcal{L}MEkeys$ contains pairs of $(ME, \mathbb{P}ESK^{I}, \mathbb{P}ESK^{II}, \mathbb{P}EPK^{I}, \mathbb{P}EPK^{II})$, where ME, $(\mathbb{P}ESK^{I}, \mathbb{P}ESK^{II})$ and $(\mathbb{P}EPK^{I}, \mathbb{P}EPK^{II})$, respectively, are presented as the information of member entity, entity secret keys and entity public keys.

– *Phase* 1: The adversary $\mathcal{A}_I$ has the capability to make at most ψ_1 queries as follows.

✓ Q_G *query*: By providing $\mathbb{B}G_{q,u,m}$, $\mathbb{B}G_{q,u,n}$ and ACT as inputs to this query, the execution of the following steps will produce a response denoted by $\mathbb{B}G_{q,u,l}$.
- Execute Rule-2 to transform $(\mathbb{B}G_{q,u,m}, \mathbb{B}G_{q,u,n})$ into $(\mathbb{P}G_{q,u,m}, \mathbb{P}G_{q,u,n})$.
- Compute $\mathbb{P}G_{q,u,l} = \mathbb{P}G_{q,u,m} + \mathbb{P}G_{q,u,n}$ if $ACT =$ "multiplication" and $\mathbb{P}G_{q,u,l} = \mathbb{P}G_{q,u,m} - \mathbb{P}G_{q,u,n}$ if $ACT =$ "division".
- Execute Rule-1 to transform $\mathbb{P}G_{q,u,l}$ into $\mathbb{B}G_{q,u,l}$.

✓ Q_{G_T} *query*: By providing $\mathbb{B}G_{T,q,u,m}$, $\mathbb{B}G_{T,q,u,n}$ and ACT as inputs to this query, the execution of the following steps will produce a response denoted by $\mathbb{B}G_{T,q,u,l}$.
- Execute Rule-2 to transform $(\mathbb{B}G_{T,q,u,m}, \mathbb{B}G_{T,q,u,n})$ into $(\mathbb{P}G_{T,q,u,m}, \mathbb{P}G_{T,q,u,n})$.
- Compute $\mathbb{P}G_{T,q,u,l} = \mathbb{P}G_{T,q,u,m} + \mathbb{P}G_{T,q,u,n}$ if $ACT =$ "multiplication" and $\mathbb{P}G_{T,q,u,l} = \mathbb{P}G_{T,q,u,m} - \mathbb{P}G_{T,q,u,n}$ if $ACT =$ "division".
- Execute Rule-1 to transform $\mathbb{P}G_{T,q,u,l}$ into $\mathbb{B}G_{T,q,u,l}$.

✓ $Q_{\hat{e}}$ *query*: By providing $\mathbb{B}G_{q,u,m}$ and $\mathbb{B}G_{q,u,n}$ as inputs to this query, the execution of the following steps will produce a response denoted by $\mathbb{B}G_{T,q,u,l}$.
- Execute Rule-2 to transform $(\mathbb{B}G_{q,u,m}, \mathbb{B}G_{q,u,n})$ into $(\mathbb{P}G_{q,u,m}, \mathbb{P}G_{q,u,n})$.
- Compute $\mathbb{P}G_{T,q,u,l} = \mathbb{P}G_{q,u,m} \cdot \mathbb{P}G_{q,u,n}$.

- Execute Rule-1 to transform $\mathbb{P}G_{T,q,u,l}$ into $\mathbb{B}G_{T,q,u,l}$.

✓ *EntityKeyGen query*: By providing member entity's information ME as inputs to this query, the challenger $\mathcal{CH}$ uses ME to search the list $\mathcal{L}MEkeys$. Once the matching pair $(\mathbb{P}ESK^I, \mathbb{P}ESK^{II}, \mathbb{P}EPK^I, \mathbb{P}EPK^{II})$ is located, $\mathcal{CH}$ transforms the pair into $(\mathbb{B}ESK^I, \mathbb{B}ESK^{II}, \mathbb{B}EPK^I, \mathbb{B}EPK^{II})$, respectively, and subsequently returns them to $\mathcal{A}_I$.

✓ *Signcryption query*: By inputting two entity secret keys $ESK^I_{S,i}, ESK^{II}_{S,i}$ of the member entity ME_S, identified as the sender, a message msg, and two entity public keys EPK^I_R, EPK^{II}_R of the member entity ME_R, identified as the receiver, the execution of the following steps will produce a ciphertext CT.

- Use ME_S and ME_R to find $(\mathbb{P}ESK^I_S, \mathbb{P}ESK^{II}_S)$ and $(\mathbb{P}EPK^I_R, \mathbb{P}EPK^{II}_R)$ in the list $\mathcal{L}MEkeys$, respectively.
- Choose the variate $\mathbb{P}H_4$ from the list $\mathcal{L}HF_4$ by using $msg||h$, where h is a random value.
- Set $\mathbb{P}U = \mathbb{P}H_4$ and pick a random variate $\mathbb{P}V$ in G.
- Choose the variate $\mathbb{P}H_2$ from the list $\mathcal{L}HF_2$ by using $\mathbb{P}EPK^I_R \cdot \mathbb{P}V||\mathbb{P}U||\mathbb{P}V$.
- Execute Rule-1 to transform $\mathbb{P}H_2$ into $\mathbb{B}H_2$.
- Compute $\mathbb{B}R = \mathbb{B}H_2 \oplus (msg||h)$, and execute Rule-2 to transform $\mathbb{B}R$ into $\mathbb{P}R$.
- Choose the variate $\mathbb{P}H_1$ from the list $\mathcal{L}HF_1$ by using $\mathbb{P}EPK^{II}_R \cdot \mathbb{P}V$, and choose the variate $\mathbb{P}H_3$ from the list $\mathcal{L}HF_3$ by using msg.
- Set $\mathbb{P}S = \mathbb{P}H_1 + \mathbb{P}H_3 \cdot \mathbb{P}U$.
- Choose the variate $\mathbb{P}H_5$ from the list $\mathcal{L}HF_5$ by using $ME_S||ME_R||\mathbb{P}U||\mathbb{P}V||\mathbb{P}R||\mathbb{P}S||msg$.
- Set $\mathbb{P}\sigma = \mathbb{P}ESK^I_S + (\mathbb{P}X + \mathbb{P}Y \cdot \mathbb{P}H_5) \cdot \mathbb{P}U + \mathbb{P}ESK^{II}_S + (\mathbb{P}X + \mathbb{P}Y \cdot \mathbb{P}H_5) \cdot \mathbb{P}V$.
- Set $CT = (ME_S, ME_R, \mathbb{P}U, \mathbb{P}V, \mathbb{P}R, \mathbb{P}S, \mathbb{P}\sigma)$.

✓ *Signcryption leak query*: By providing an index i, and two leaked functions $LF^A_{SC,i}$ and $LF^B_{SC,i}$ as inputs to this query, the challenger $\mathcal{CH}$ provides two sets of leaked information $\Lambda LF^A_{SC,i} = LF^A_{SC,i}(ESK^I_{S,i,A}, ESK^{II}_{S,i,A})$ and $\Lambda LF^B_{SC,i} = LF^B_{SC,i}(ESK^I_{S,i,B}, ESK^{II}_{S,i,B})$ back to $\mathcal{A}_I$.

✓ *Unsigncryption query*: By inputting two entity secret keys $ESK^I_{R,j}, ESK^{II}_{R,j}$ of the member entity ME_R, identified as the receiver, a ciphertext CT message msg, and two entity public keys EPK^I_S, EPK^{II}_S of the member entity ME_S, identified as the sender, the execution of the following steps will produce a message msg.

- Choose the variate $\mathbb{P}H_2$ from the list $\mathcal{L}HF_2$ by using $\mathbb{P}ESK^I_R \cdot \mathbb{P}V||\mathbb{P}U||\mathbb{P}V$.
- Execute Rule-1 to transform $\mathbb{P}H_2$ and $\mathbb{P}R$ into $\mathbb{B}H_2$ and $\mathbb{B}R$.
- Compute $\mathbb{B}R \oplus \mathbb{B}H_2$, and obtain msg' and h'.
- Choose the variate $\mathbb{P}H_1$ from the list $\mathcal{L}HF_1$ by using $\mathbb{P}ESK^{II}_R \cdot \mathbb{P}V$.
- Choose the variate $\mathbb{P}H_3$ from the list $\mathcal{L}HF_3$ by using msg'.
- Choose the variate $\mathbb{P}H_4$ from the list $\mathcal{L}HF_4$ by using $msg'||h'$.
- If both equations $\mathbb{P}U = \mathbb{P}H_4$ and $\mathbb{P}S = \mathbb{P}H_1 + \mathbb{P}H_3 \cdot \mathbb{P}H_4$ hold, find $\mathbb{P}H_5$ from the list $\mathcal{L}HF_5$ by using $ME_S||ME_R||\mathbb{P}U||\mathbb{P}V||\mathbb{P}R||\mathbb{P}S||msg'$.
- If $\mathbb{P}g \cdot \mathbb{P}\sigma = \mathbb{P}EPK^I_S + \mathbb{P}EPK^{II}_S + (\mathbb{P}U + \mathbb{P}V) \cdot (\mathbb{P}X + \mathbb{P}Y \cdot \mathbb{P}H_5)$ holds, return the message msg'.

- ✓ *Unsigncryption leak query*: By providing an index j, and two leaked functions $LF^{A}_{USC,j}$ and $LF^{B}_{USC,j}$ as inputs to this query, the challenger $\mathcal{CH}$ provides two sets of leaked information $\Lambda LF^{A}_{USC,j} = LF^{A}_{USC,j}(ESK^{I}_{R,j,A}, ESK^{II}_{R,j,A})$ and $\Lambda LF^{B}_{USC,j} = LF^{B}_{USC,j}(ESK^{I}_{R,j,B}, ESK^{II}_{R,j,B})$ back to $\mathcal{A}_I$.
- ✓ *Authorization query*: By providing member entity's information ME as inputs to this query, the challenger $\mathcal{CH}$ uses ME to search the list $\mathcal{L}MEkeys$. Once the matching pair $(\mathbb{P}ESK^{I}, \mathbb{P}ESK^{II}, \mathbb{P}EPK^{I}, \mathbb{P}EPK^{II})$ is located, $\mathcal{CH}$ transforms the pair into $(\mathbb{B}ESK^{I}, \mathbb{B}ESK^{II}, \mathbb{B}EPK^{I}, \mathbb{B}EPK^{II})$ respectively. Then, $\mathcal{CH}$ sets $TD = \mathbb{B}ESK^{II}$, and subsequently returns it to $\mathcal{A}_I$.
- ✓ *Authorization leak query*: By providing an index k, and two leaked functions $LF^{A}_{Auth,k}$ and $LF^{B}_{Auth,k}$ as inputs to this query, the challenger $\mathcal{CH}$ provides two sets of leaked information $\Lambda LF^{A}_{Auth,k} = LF^{A}_{Auth,k}(ESK^{II}_{R,j,A})$ and $\Lambda LF^{B}_{Auth,k} = LF^{B}_{Auth,k}(ESK^{II}_{R,j,B})$ back to $\mathcal{A}_I$.

– *Challenger*: $\mathcal{A}_I$ chooses a specific member entity ME^{*}_{R}, and provides two target plaintexts, msg_0 and msg_1, to $\mathcal{CH}$. Subsequently, $\mathcal{CH}$ chooses a random bit $b \in \{0, 1\}$ and utilizes the *Signcryption* algorithm with the corresponding parameters msg^{*}_{b}, EPK^{I}_{R}, EPK^{II}_{R}, $ESK^{I}_{S,i}$ and $ESK^{II}_{S,i}$ to generate the target ciphertext CT^{*}. Then, $\mathcal{CH}$ sends the resulting ciphertext CT^{*} to $\mathcal{A}_I$.
– *Phase* 2: The adversary $\mathcal{A}_I$ can make further queries at most ψ_2 times as in the *Phase* 1 except that the selected target, namely ME^{*}_{R}, msg^{*}_{b}, $ESK^{I}_{S,i}$ and $ESK^{II}_{S,i}$, may not appear in the *EntityKeyGen*, the *Unsigncryption* and the *Authorization queries*.
– *Guess phase*: $\mathcal{A}_I$ produces the value b' and succeeds in the game if b' is equal to b. The advantage of winning the game is represented by $Adv(\mathcal{A}_I) = |\Pr[b' = b]–1/2|$.

In the following, we explore the advantage that $\mathcal{A}_I$ wins the security game. We discuss $\mathcal{A}_I$'s advantage in two scenarios: (1) $\mathcal{A}_I$ refrains from employing leak queries; (2) $\mathcal{A}_I$ utilizes *Signcryption leak query*, *Unsigncryption leak query* and *Authorization leak query*.

– Scenario I: $\mathcal{A}_I$ possesses $Adv^{nlq}_{\mathcal{A}_I} \leqq |\Pr[\text{Case 1}] + \Pr[\text{Case 2}] - 1/2| \leqq |384(\psi_1 + \psi_2)^2/q + 1/2 - 1/2| = O((\psi_1 + \psi_2)^2/q)$ in winning the security game, where Pr[Case 1] and Pr[Case 2] are described below.
  - ✓ Pr[Case 1] refers to the probability of encountering a collision in either $\mathcal{L}G$ or $\mathcal{L}G_T$. Let's focus on the collision probability within the list $\mathcal{L}G$. Assume that we have j elements in $\mathcal{L}G$, denoted by $v_i \in Z^{*}_{q}$ for $i \in [1, j]$, where j random values are considered. In the event of a collision, we observe $\mathbb{P}G(v_1, v_2, \ldots, v_j) = \mathbb{P}G_m - \mathbb{P}G_n = 0$, where $\mathbb{P}G_m$ and $\mathbb{P}G_n$ represent any two polynomials from the list $\mathcal{L}G$. This implies that $\mathbb{P}G_m$ and $\mathbb{P}G_n$ have the same value, which resembles a collision in a hash function. Therefore, if one could efficiently find such a collision, it would imply the ability to solve the discrete logarithm problem. Utilizing Lemma 2, we can evaluate the probability of a collision occurring within $\mathcal{L}G$, estimated as $(3/q)\binom{|\mathcal{L}G|}{2}$, based on the fact that the maximum degree of a polynomial in $\mathcal{L}G$ is 3, as observed from the *Signcryption query* where the highest-degree term $\mathbb{P}Y \cdot \mathbb{P}H_5 \cdot \mathbb{P}U$ has degree 3. Similarly, the probability of a collision occurring within the list $\mathcal{L}G_T$ is $(6/q)\binom{|\mathcal{L}G_T|}{2}$. Thus, we have Pr[Case 1] $\leqq (3/q)\binom{|\mathcal{L}G|}{2} + (6/q)\binom{|\mathcal{L}G_T|}{2} \leqq (6/q)(|\mathcal{L}G| + |\mathcal{L}G_T|)^2 \leqq 384(\psi_1 + \psi_2)^2/q$ due to the fact that $|\mathcal{L}G| + |\mathcal{L}G_T| \leqq 8(\psi_1 + \psi_2)$.

✓ Pr[Case 2] refers to the probability of encountering a correct guess $b' = b$ and so Pr[Case 2] $\leqq 1/2$.

– Scenario II: $\mathcal{A}_I$ possesses $Adv^{lq}_{\mathcal{A}_I} \leqq O((\psi_1+\psi_2)^2/q \cdot 2^{2\Phi}) + O((\psi_1+\psi_2)^2/q) = O((\psi_1+\psi_2)^2/q \cdot 2^{2\Phi})$ in successfully winning the security game by the following cases.

✓ $\mathcal{A}_I$ issues the *Signcryption leak query*: Through this query, $\mathcal{A}_I$ acquires $\Lambda LF^A_{SC,i}$ and $\Lambda LF^B_{SC,i}$ from two leaked functions $LF^A_{SC,i}$ and $LF^B_{SC,i}$, where $\Lambda LF^A_{SC,i}$ is defined as $LF^A_{SC,i}(ESK^I_{S,i,A}, ESK^{II}_{S,i,A})$, and $\Lambda LF^B_{SC,i}$ corresponds to $LF^B_{SC,i}(ESK^I_{S,i,B}, ESK^{II}_{S,i,B})$. Here, ESK^I_S and ESK^{II}_S can be obtained from the equations $ESK^I_{S,0,A} \cdot ESK^I_{S,0,B} = ESK^I_{S,1,A} \cdot ESK^I_{S,1,B} = \cdots = ESK^I_{S,i-1,A} \cdot ESK^I_{S,i-1,B} = ESK^I_{S,i,A} \cdot ESK^I_{S,i,B}$, and $ESK^{II}_{S,0,A} \cdot ESK^{II}_{S,0,B} = ESK^{II}_{S,1,A} \cdot ESK^{II}_{S,1,B} = \cdots = ESK^{II}_{S,i-1,A} \cdot ESK^{II}_{S,i-1,B} = ESK^{II}_{S,i,A} \cdot ESK^{II}_{S,i,B}$. By employing key update techniques in Galindo and Virek (2013) with the constraint that $|\Lambda LF^A_{SC,i}|$ and $|\Lambda LF^B_{SC,i}|$ are both less than Φ, the adversary $\mathcal{A}_I$'s ability is restricted to acquiring a maximum of 2Φ bits of ESK^I_S and ESK^{II}_S.

✓ $\mathcal{A}_I$ issues the *Unsigncryption leak query*: Through this query, $\mathcal{A}_I$ acquires $\Lambda LF^A_{USC,j}$ and $\Lambda LF^B_{USC,j}$ from two leaked functions $LF^A_{USC,j}$ and $LF^B_{USC,j}$, where $\Lambda LF^A_{USC,j}$ is defined as $LF^A_{USC,j}(ESK^I_{R,j,A}, ESK^{II}_{R,j,A})$, and $\Lambda LF^B_{USC,j}$ corresponds to $LF^B_{USC,j}(ESK^I_{R,j,B}, ESK^{II}_{R,j,B})$. Here, ESK^I_R and ESK^{II}_R can be obtained from the equations $ESK^I_{R,0,A} \cdot ESK^I_{R,0,B} = ESK^I_{R,1,A} \cdot ESK^I_{R,1,B} = \cdots = ESK^I_{R,j-1,A} \cdot ESK^I_{R,j-1,B} = ESK^I_{R,j,A} \cdot ESK^I_{R,j,B}$, and $ESK^{II}_{R,0,A} \cdot ESK^{II}_{R,0,B} = ESK^{II}_{R,1,A} \cdot ESK^{II}_{R,1,B} = \cdots = ESK^{II}_{R,j-1,A} \cdot ESK^{II}_{R,j-1,B} = ESK^{II}_{R,j,A} \cdot ESK^{II}_{R,j,B}$. By employing key update techniques in Galindo and Virek (2013) with the constraint that $|\Lambda LF^A_{USC,j}|$ and $|\Lambda LF^B_{USC,j}|$ are both less than Φ, the adversary $\mathcal{A}_I$'s ability is restricted to acquiring a maximum of 2Φ bits of ESK^I_R and ESK^{II}_R.

✓ $\mathcal{A}_I$ issues the *Authorization leak query*: Through this query, $\mathcal{A}_I$ acquires $\Lambda LF^A_{Auth,k}$ and $\Lambda LF^B_{Auth,k}$ from two leaked functions $LF^A_{Auth,k}$ and $LF^B_{Auth,k}$, where $\Lambda LF^A_{Auth,k}$ is defined as $LF^A_{Auth,k}(ESK^{II}_{R,j,A})$, and $\Lambda LF^B_{Auth,k}$ corresponds to $LF^B_{USC,j}(ESK^{II}_{R,j,B})$. Here, ESK^{II}_R can be obtained from the equation $ESK^{II}_{R,0,A} \cdot ESK^{II}_{R,0,B} = ESK^{II}_{R,1,A} \cdot ESK^{II}_{R,1,B} = \cdots = ESK^{II}_{R,k-1,A} \cdot ESK^{II}_{R,k-1,B} = ESK^{II}_{R,k,A} \cdot ESK^{II}_{R,k,B}$. By employing key update techniques in Galindo and Virek (2013) with the constraint that $|\Lambda LF^A_{Auth,k}|$ and $|\Lambda LF^B_{Auth,k}|$ are both less than Φ, the adversary $\mathcal{A}_I$'s ability is restricted to acquiring a maximum of 2Φ bits of ESK^{II}_R.

Based on the aforementioned discussions, three events are defined as follows:

– In the first event $\mathcal{E}ESK^I$, $\mathcal{A}_I$ has the capability to derive ESK^I from $\Lambda LF^A_{SC,i}$, $\Lambda LF^B_{SC,i}$, $\Lambda LF^A_{USC,j}$ and $\Lambda LF^B_{USC,j}$. Furthermore, the complementary event of $\mathcal{E}ESK^I$ is denoted as $\overline{\mathcal{E}ESK^I}$.

– In the second event $\mathcal{E}ESK^{II}$, $\mathcal{A}_I$ has the capability to derive ESK^{II} from $\Lambda LF^A_{SC,i}$, $\Lambda LF^B_{SC,i}$, $\Lambda LF^A_{USC,j}$, $\Lambda LF^B_{USC,j}$, $\Lambda LF^A_{Auth,k}$ and $\Lambda LF^B_{Auth,k}$. Furthermore, the complementary event of $\mathcal{E}ESK^{II}$ is denoted as $\overline{\mathcal{E}ESK^{II}}$.

– In the third event $\mathcal{E}CG$, $\mathcal{A}_I$ has a correct guess.

Considering these three events, we can compute the probability $\Pr[\mathcal{A}_I]$ of $\mathcal{A}_I$ winning this game.

$$\begin{aligned}\Pr[\mathcal{A}_I] &= \Pr[\mathcal{ECG}] \\ &= \Pr\left[\mathcal{ECG} \wedge \left(\mathcal{ESK}^I \vee \mathcal{ESK}^{II}\right)\right] + \Pr\left[\mathcal{ECG} \wedge \left(\overline{\mathcal{ESK}^I} \wedge \overline{\mathcal{ESK}^{II}}\right)\right] \\ &\leqq \Pr\left[\mathcal{ESK}^I \vee \mathcal{ESK}^{II}\right] + \Pr\left[\mathcal{ECG} \wedge \left(\overline{\mathcal{ESK}^I} \wedge \overline{\mathcal{ESK}^{II}}\right)\right]\end{aligned}$$

Because of Lemma 2, we obtain $\Pr[\mathcal{ESK}^I \wedge \mathcal{ESK}^{II}] \leqq Adv_{\mathcal{A}_I}^{nlq} \cdot 2^{2\Phi} \leqq O((\psi_1 + \psi_2)^2/q \cdot 2^{2\Phi})$. Since $\Pr[\mathcal{ECG} \wedge (\overline{\mathcal{ESK}^I} \wedge \overline{\mathcal{ESK}^{II}})]$ indicates the advantage of correct guess while having no knowledge of ESK^I and ESK^{II}, we have $\Pr[\mathcal{ECG} \wedge (\overline{\mathcal{ESK}^I} \wedge \overline{\mathcal{ESK}^{II}})] = Adv\mathcal{A}_I{}^{nlq} = O((\psi_1 + \psi_2)^2/q)$. Finally, we have $\Pr[\mathcal{A}_I] = \Pr[\mathcal{ECG}] \leqq O((\psi_1 + \psi_2)^2/q \cdot 2^{2\Phi}) + O((\psi_1 + \psi_2)^2/q) = O((\psi_1 + \psi_2)^2/q \cdot 2^{2\Phi})$. □

Theorem 2. *Under the assumptions of DL and HF, the proposed LR-PKSCET scheme possesses leakage resilience and one-way security in the security game* (*Definition* 3) *using the GBG model.*

Proof. Let's begin the security game with the interaction between a challenger $\mathcal{CH}$ and an adversary $\mathcal{A}_{II}$.

- *Setup*: This stage is the same as that in the proof of Theorem 1.
- *Phase* 1: This stage is the same as that in the proof of Theorem 1.
- *Challenge*: $\mathcal{A}_{II}$ chooses a specific member entity ME_R^* to $\mathcal{CH}$. Subsequently, $\mathcal{CH}$ chooses a random message msg^* and utilizes the *Signcryption* algorithm with the corresponding parameters msg^*, EPK_R^I, EPK_R^{II}, ESK_S^I and ESK_S^{II} to generate the target ciphertext CT^*. Then, $\mathcal{CH}$ sends CT^* to $\mathcal{A}_{II}$.
- *Phase* 2: The adversary $\mathcal{A}_{II}$ can make further queries at most ψ_2 times as in the *Phase* 1 except that the selected target, namely ME_R^*, msg^*, $ESK_{S,i}^I$ and $ESK_{S,i}^{II}$, may not appear in the *EntityKeyGen* and the *Unsigncryption* queries.
- *Guess phase*: $\mathcal{A}_{II}$ produces the message msg' and wins the game if msg' is the same as msg. The advantage of winning the game is represented by $Adv(\mathcal{A}_{II}) = |\Pr[msg' = msg]|$.

In the following, we explore the advantage that $\mathcal{A}_{II}$ wins in the security game. We discuss $\mathcal{A}_{II}$'s advantage in two scenarios: (1) $\mathcal{A}_{II}$ refrains from employing leak queries; (2) $\mathcal{A}_{II}$ utilizes *Signcryption leak query* and *Unsigncryption leak query*.

- Scenario I: $\mathcal{A}_{II}$ possesses $Adv_{A_{II}}^{nlq} \leqq |\Pr[\text{Case 1}] + \Pr[\text{Case 2}]| \leqq |384(\psi_1 + \psi_2)^2/q + 2/q| = O((\psi_1 + \psi_2)^2/q)$ in winning the security game, where Pr[Case 1] and Pr[Case 2] are described below.
  - ✓ Pr[Case 1] refers to the probability of encountering a collision in either $\mathcal{L}G$ or $\mathcal{L}G_T$. We can obtain $\Pr[\text{Case 1}] \leqq 384(\psi_1 + \psi_2)^2/q$ using a proof similar to that in the proof of Theorem 1.

- ✓ Pr[Case 2] refers to the probability of encountering a correct output $msg' = msg$. Certainly, $\mathcal{A}_{II}$ will be provided with a target ciphertext $CT^* = (ME_S^*, ME_R^*, U^*, V^*, R^*, S^*, \sigma^*)$, and $\mathcal{A}_{II}$ will utilize the *Unsigncryption* algorithm to derive the message msg'. Notably, the message msg' is computed using the expression $R^* \oplus HF_2(\hat{e}(ESK_R^I, V^*), U^*, V^*)$. Let's denote $\mathbb{P}T = \mathbb{P}V^* \cdot \mathbb{P}ESK_R^I$. The polynomial $\mathbb{P}T$ exhibits a maximum degree of 2. By Lemma 2, the probability Pr[Case 2] $\leqq 2/q$ can be achieved.

– Scenario II: By employing a similar approach to the proof of Theorem 1, we have that $\mathcal{A}_{II}$ possesses $Adv_{\mathcal{A}_{II}}^{1q} \leqq O((\psi_1 + \psi_2)^2/q \cdot 2^{2\Phi}) + O((\psi_1 + \psi_2)^2/q) = O((\psi_1 + \psi_2)^2/q \cdot 2^{2\Phi})$ in winning the security game. □

Theorem 3. *Under the assumptions of DL and HF, the proposed LR-PKSCET scheme possesses leakage resilience and existential unforgeability in the security game* (*Definition 4*) *using the GBG model.*

Proof. Let's begin the security game with the interaction between a challenger $\mathcal{CH}$ and an adversary $\mathcal{A}_{III}$.

– *Setup*: This stage is the same as that in the proof of Theorem 1.
– *Phase* 1: This stage is the same as that in the proof of Theorem 1.
– *Forgery*: The adversary $\mathcal{A}_{III}$ successfully forges a ciphertext $CT^* = (ME_S^*, ME_R^*, U^*, V^*, R^*, S^*, \sigma^*)$ for a message msg^*, and we declare $\mathcal{A}_{III}$ as the winner of this game if the following conditions are satisfied.
  - ✓ The *Unsigncryption* algorithm is capable of generating the message msg^*.
  - ✓ The *Signcryption queries* do not include the message msg^*, and that also do not contain the two member entities ME_S^* and ME_R^*.
  - ✓ The *EntityKeyGen queries* do not include the member entity ME_S^*.

In the following, we explore the advantage that $\mathcal{A}_{III}$ wins the security game. We discuss $\mathcal{A}_{III}$'s advantage in two scenarios: (1) $\mathcal{A}_{III}$ refrains from employing leak queries; (2) $\mathcal{A}_{III}$ utilizes *Signcryption leak query*, *Unsigncryption leak query* and *Authorization leak query*.

– Scenario I: $\mathcal{A}_{III}$ possesses $Adv_{\mathcal{A}_{III}}^{nlq} \leqq |\text{Pr[Case 1]} + \text{Pr[Case 2]}| \leqq |384\psi_1^2/q + 3/q| = O(\psi_1^2/q)$ in winning the security game, where Pr[Case 1] and Pr[Case 2] are described below.
  - ✓ Pr[Case 1] refers to the probability of encountering a collision in either $\mathcal{L}G$ or $\mathcal{L}G_T$. We can obtain Pr[Case 1] $\leqq 384(\psi_1 + \psi_2)^2/q$ using a proof similar to that in the proof of Theorem 1.
  - ✓ Pr[Case 2] refers to the probability of forging a valid pair $(msg^*, CT^* = (ME_S^*, ME_R^*, U^*, V^*, R^*, S^*, \sigma^*))$. The valid pair satisfies $\mathbb{P}g \cdot \mathbb{P}\sigma = \mathbb{P}EPK_S^I + \mathbb{P}EPK_S^{II} + (\mathbb{P}U + \mathbb{P}V) \cdot (\mathbb{P}X + \mathbb{P}Y \cdot \mathbb{P}H_5)$ in the *Unsigncryption* algorithm. Let's denote $\mathbb{P}T = \mathbb{P}g \cdot \mathbb{P}\sigma - \mathbb{P}EPK_S^I + \mathbb{P}EPK_S^{II} + (\mathbb{P}U + \mathbb{P}V) \cdot (\mathbb{P}X + \mathbb{P}Y \cdot \mathbb{P}H_5)$. The polynomial $\mathbb{P}T$ exhibits a maximum degree of 3. By Lemma 2, the probability Pr[Case 2] $\leqq 3/q$ can be achieved.

– Scenario II: By employing a similar approach to the proof of Theorem 1, we have that $\mathcal{A}_{III}$ possesses $Adv^{lq}_{\mathcal{A}_{III}} \leqq O(\psi_1^2/q \cdot 2^{2\Phi}) + O(\psi_1^2/q) = O(\psi_1^2/q \cdot 2^{2\Phi})$ in winning the security game. □

7. Comparisons

We compare the proposed LR-PKSCET scheme with the existing PKEET scheme (Ma *et al.*, 2015), PKSCET scheme (Le *et al.*, 2021), LR-PKE scheme (Galindo *et al.*, 2016), and LR-PKSC scheme (Tseng *et al.*, 2022). Table 2 summarizes these comparisons on three properties: possession of signature and encryption, permission of secret keys leakage, and with the property of equality test. Firstly, let's consider the PKEET scheme (Ma *et al.*, 2015), which focuses on the equality test functionality, but falls short in the other two properties. On the other hand, the PKSCET scheme (Le *et al.*, 2021) possesses the property of performing both signature and encryption, along with the desirable equality test functionality. Unfortunately, it does not allow secret keys leakage. Conversely, the LR-PKE scheme (Galindo *et al.*, 2016) addresses the issue of secret keys leakage, but does so without possessing both signature and encryption capabilities or the equality test functionality. In contrast, the LR-PKSC scheme (Tseng *et al.*, 2022) combines the advantage of possessing both signature and encryption capabilities along with the ability to allow secret keys leakage. However, it falls short in providing the equality test functionality. In conclusion, the proposed LR-PKSCET emerges as a versatile solution that includes all three properties – having signature and encryption capabilities, permission of secret key leakage, and with the equality test functionality.

Next, we introduce a pair of symbols aimed at evaluating the computational effort of the algorithms of our LR-PKSCET scheme:

- CE_{bp}: This symbol denotes the computational effort required for performing a bilinear pairing operation $\hat{e} : G \times G \rightarrow G_T$.
- CE_{exp}: This symbol denotes the computational effort required for performing exponentiation of the group G or G_T.

We derive the pertinent result from simulations (Xiong and Qin, 2015) to obtain an approximate 20 ms for CE_{bp} and 7 ms for CE_{exp}. These simulations are conducted under a computer environment of an Intel Core i7 CPU with 1.80 GHz. The input of simulations

Table 2
Comparison of our LR-PKSCET with existing PKEET, PKSCET, LR-PKE and LR-PKSC.

Schemes	Possession of signature and encryption	Permission of secret keys leakage	With the property of equality test
Ma *et al.*'s PKEET scheme (2015)	No	No	Yes
Le *et al.*'s PKSCET scheme (2021)	Yes	No	Yes
Galindo *et al.*'s LR-PKE scheme (2016)	No	Yes	No
Tseng *et al.*'s LR-PKSC scheme (2022)	Yes	Yes	No
Our LR-PKSCET scheme	Yes	Yes	Yes

Table 3
Computational cost of our LR-PKSCET.

Algorithms	Computational effort	Running time on a MD	Running time on a PC
Initialization	$2\,CE_{exp}$	62 ms	14 ms
EntityKeyGen	$2\,CE_{bp} + 6CE_{exp}$	378 ms	82 ms
Signcryption	$10\,CE_{exp}$	310 ms	70 ms
Unsigncryption	$5\,CE_{bp} + 5CE_{exp}$	635 ms	135 ms
Authorization	$2\,CE_{exp}$	62 ms	14 ms
Test	$4\,CE_{bp}$	384 ms	80 ms

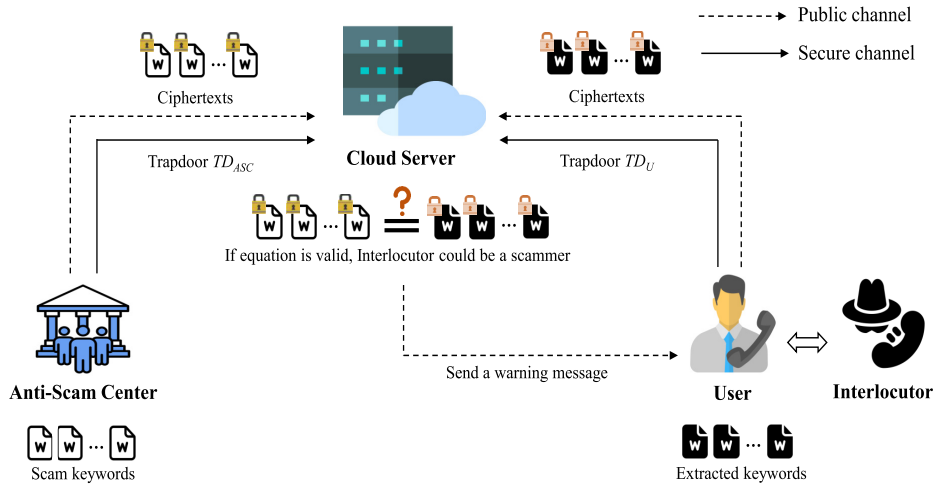

Fig. 3. Building an anti-scam system based on the proposed LR-PKSCET scheme.

encompasses a finite field denoted as F_q, along with the groups G and G_T. Here, q denotes a prime number with 512 bits, and it concurrently serves as the order of the two groups G and G_T. The simulations involve a mobile device environment that employs an Intel 624-MHz PXA270 CPU. We obtain CE_{bp} and CE_{exp} to be approximately 96 ms and 31 ms, respectively. For a more comprehensive understanding, Table 3 presents an overview of the computational effort of our LR-PKSCET scheme linked to distinct algorithmic processes, including *Initialization*, *EntityKeyGen*, *Signcryption*, *Unsigncryption*, *Authorization* and *Test* on a mobile device MD and a PC.

8. Application

As mentioned in Section 1, our LR-PKSCET scheme can be applied to anti-scam systems. As depicted in Fig. 3, we present a situation that exemplifies the utilization of the LR-PKSCET scheme to counteract telephone scams. The scenario encompasses four key roles: anti-scam centre, interlocutor, user, and cloud server. Now, we provide a breakdown

of the responsibilities and functions associated with each role within the presented scenario.

- ✓ The anti-scam centre (*ASC*) continuously collects keywords related to telephone scams and subjects these keywords to a signcryption process (by using *Signcryption* algorithm). Also, the *ASC* employs its own secret key to perform *Authorization* algorithm to generate a trapdoor TD_{ACS}. Subsequently, the *ASC* transmits the generated ciphertexts and the trapdoor TD_{ACS} to the cloud server through public and secure channels, respectively.
- ✓ The interlocutor could potentially be a scammer, who initiates a call to the user with the intention of orchestrating a fraudulent scheme over the phone.
- ✓ The user is vulnerable to potential scammers. When an interlocutor's call is received, the mobile application automatically transforms the spoken content into text. Relevant keywords are extracted from this transcribed text, and then these keywords will also be subjected to the signcryption procedure. On the other hand, the user also employs its own secret key to generate a trapdoor TD_U. Subsequently, the user transmits the generated ciphertexts and the trapdoor TD_U to the cloud server through public and secure channels, respectively.
- ✓ The cloud server (*CS*) is responsible for performing equality tests of ciphertexts using the *Test* algorithm, which determines whether two ciphertexts contain the same plaintext. During this test process, the *CS* remains unaware of the actual plaintext content. Upon detecting a specific number of matches, the *CS* will promptly dispatch a warning message to the user. This message serves to alert the user that the ongoing conversation could be linked to a scam activity. As a result, the user may be able to avoid this scam activity.

According to the scenario described above, the proposed LR-PKSCET scheme demonstrates a collaborative approach to counter telephone scams effectively. The synergy among the *ASC*, interlocutor, user, and *CS* displays the potential to enhance security measures in the realm of telecommunication.

9. Conclusions and Future Work

In this paper, we have presented a novel solution to the critical challenge of enhancing the security of PKSCET against side-channel attacks. Our proposed leakage-resilient public key signcryption with equality test (LR-PKSCET) scheme not only successfully combines the benefits of public key signcryption and equality test properties but also offers robust resistance against side-channel attacks. Through rigorous analysis and security proofs, we have demonstrated that the LR-PKSCET scheme achieves several essential security attributes, including leakage resilience, indistinguishability, one-wayness, and existential unforgeability. By incorporating the proposed LR-PKSCET scheme into an anti-scam system, we offer a practical application that addresses a pressing societal concern. The integration of our scheme into such a system has the potential to significantly reduce the frequency and impact of scam cases, thereby safeguarding users from financial and

personal losses. While our proposed scheme is based on classical bilinear pairing and achieves indistinguishable chosen-ciphertext attacks (IND-CCA) security, we recognize the increasing importance of post-quantum cryptography (PQC). Although some research efforts have been made in designing PQC-based public key signcryption with equality test (PKSCET), these schemes still lack the functionality to side-channel attacks. As part of our future work, we aim to design a PQC-based LR-PKSCET scheme that maintains IND-CCA security and addresses the challenges posed by the quantum era.

Funding

This research was partially supported by National Science and Technology Council, Taiwan, under contract no. NSTC112-2634-F-027-001-MBK.

References

Agrawal, S., Boneh, D., Boyen, X. (2010). Efficient lattice (h)ibe in the standard model. In: *LNCS*, Vol. 6110. *EUROCRYPT'10*, pp. 553–572.

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

Alouffi, B., Hasnain, M., Alharbi, A., Alosaimi, W., Alyami, H., Ayaz, M. (2021). A systematic literature review on cloud computing security: Threats and mitigation strategies. *IEEE Access*, 9, 57792–57807.

Boneh, D., Crescenz, G.D., Ostrovsky, R., Persiano, G. (2004). Public key encryption with keyword search. In: *EUROCRYPT'04*, *LNCS*, Vol. 3027. pp. 506–522.

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.

Deverajan, G.G., Muthukumaran, V., Hsu, C.-H., Karuppiah, M., Chung, Y.-C., Chen, Y.-H. (2021). Public key encryption with equality test for industrial Internet of Things system in cloud computing. *Transactions on Emerging Telecommunications Technologies*, 4, e4202.

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.

Duong, D.H., Fukushima, K., Kiyomoto, S., Roy, P.S., Susilo, W. (2019). A lattice-based public key encryption with equality test in standard model. In: *ACISP'19*, *LNCS*, Vol. 11547. pp. 138–155.

Dziembowski, S., Pietrzak, K. (2008). Leakage-resilient cryptography. In: *FOCS'08*, 293–302.

Faust, S., Kiltz, E., Pietrzak, K., Rothblum, G. (2010). Leakage-resilient signatures. In: *TCC'10*, *LNCS*, Vol. 5978. pp. 343–360.

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.

Galindo, D., Virek, S. (2013). A practical leakage-resilient signature scheme in the generic group model. In: *SAC'13*, *LNCS*, Vol. 7707. pp. 50–65.

Huang, K., Tso, R., Chen, Y.-C., Li, W., Sun, H. (2014). A new public key encryption with equality test. In: *NSS'15*, *LNCS*, Vol. 8792. pp. 550–557.

Huang, K., Tso, R., Chen, Y.-C., Rahman, S.M.M., Almogren, A., Alamri, A. (2015). PKE-AET: Public key encryption with authorized equality test. *The Computer Journal*, 58(10), 2686–2697.

Huang, M., Yang, B., Zhou, Y., Hu, X. (2022). Continual leakage-resilient hedged public-key encryption. *The Computer Journal*, 65(6), 1574–1585.

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

Kubota, T., Yoshida, K., Shiozaki, M., Fujino, T. (2021). Deep learning side-channel attack against hardware implementations of AES. *Microprocessors and Microsystems*, 87, 103383.

Le, H.Q., Duong, D.H., Roy, P.S., Susilo, W., Fukushima, K., Kiyomoto, S. (2021). Lattice-based signcryption with equality test in standard model. *Computer Standards & Interfaces*, 76, 103515.

Lee, H.T., Ling, S., Seo, J.H., Wang, H. (2019). Public key encryption with equality test from generic assumptions in the random oracle model. *Information Sciences*, 500, 15–33.

Lee, H.T., Ling, S., Seo, J.H., Wang, H., Youn, T. (2020). Public key encryption with equality test in the standard model. *Information Sciences*, 516, 89–108.

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

Ma, S., Huang, Q., Zhang, M., Yang, B. (2015). Efficient public key encryption with equality test supporting flexible authorization. *IEEE Transactions on Information Forensics and Security*, 10(3), 458–470.

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.

Ngo, K., Dubrova, E., Guo, Q., Johansson, T. (2021). A side-channel attack on a masked ind-CCA secure saber kem implementation. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(4), 676–707.

Sun, P. (2020). Security and privacy protection in cloud computing: Discussions and challenges. *Journal of Network and Computer Applications*, 160, 102642.

Tang, Q. (2011). Towards public key encryption scheme supporting equality test with fine-grained authorization. In: *ACISP'11*, *LNCS*, Vol. 6812, pp. 389–406.

Tang, Q. (2012a). Public key encryption supporting plaintext equality test and user-specified authorization. *Security and Communication Networks*, 5(12), 1351–1362.

Tang, Q. (2012b). Public key encryption schemes supporting equality test with authorisation of different granularity. *International Journal of Applied Cryptography*, 2(4), 304–321.

Tseng, Y.M., Tsai, T.T., Huang, S.S. (2022). Practical leakage-resilient signcryption scheme suitable for mobile environments. In: *IEEE GCCE'22*, pp. 383–384.

Xiong, H., Qin, Z. (2015). Revocable and scalable certificateless remote authentication protocol with anonymity for wireless body area networks. *IEEE Transactions on Information Forensics and Security*, 10(7), 1442–1455.

Yang, G., Tan, C.H., Huang, Q., Wong, D.S. (2010). Probabilistic public key encryption with equality test. In: *CT-RSA'10*, *LNCS*, Vol. 5985. pp. 119–131.

Zhou, Y., Hu, Z., Li, F. (2021). Searchable public-key encryption with cryptographic reverse firewalls for cloud storage. *IEEE Transactions on Cloud Computing*, 11(1), 383–396.

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