The Journal of Systems and Software 116 (2016) 101–112

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Certificate-based encryption resilient to key leakage

Qihong Yu[a], Jiguo Li[a,*], Yichen Zhang[a], Wei Wu[b,c], Xinyi Huang[b,c], Yang Xiang[c]

[a] *College of Computer and Information, Hohai University, Nanjing 211100, China*
[b] *School of Mathematics and Computer Science, Fujian Normal University, Fuzhou 350117, China*
[c] *School of Information Technology, Deakin University, Burwood, VIC 3125, Australia*

ARTICLE INFO

Article history:
Received 2 October 2014
Revised 13 April 2015
Accepted 31 May 2015
Available online 22 June 2015

Keywords:
Certificate-based encryption
Master secret key leakage
Dual system encryption

ABSTRACT

Certificate-based encryption (CBE) is an important class of public key encryption but the existing schemes are secure only under the premise that the decryption key (or private key) and master secret key are absolutely secret. In fact, a lot of side channel attacks and cold boot attacks can leak secret information of a cryptographic system. In this case, the security of the cryptographic system is destroyed, so a new model called leakage-resilient (LR) cryptography is introduced to solve this problem. While some traditional public key encryption and identity-based encryption with resilient-leakage schemes have been constructed, as far as we know, there is no leakage-resilient scheme in certificate-based cryptosystems. This paper puts forward the first certificate-based encryption scheme which can resist not only the decryption key leakage but also the master secret key leakage. Based on composite order bilinear group assumption, the security of the scheme is proved by using dual system encryption. The relative leakage rate of key is close to 1/3.

1. Introduction

In order to solve certificate management problem in traditional public key cryptosystems and the key escrow problem in identity based cryptosystems, Gentry (2003) proposed a new cryptography paradigm called certificate-based encryption. From then on, many concrete schemes (Li et al., 2010, 2012a, 2012b, 2012c, 2013; Lu and Li, 2010, 2012) were constructed under the assumption that the decryption key and master secret key are absolutely confidential.

But that is not always the case, and some side channel attacks (Halderman et al., 2009; Dodis and Pietrzak, 2010; Brumley and Boneh, 2005; Gandolfi et al., 2001; Chen et al., 2013) have been found in real world. From the attacks, the adversary can obtain some information by observing execution timing, energy consumption, etc. This results in secret information leakage which includes the information of the vital master secret key and decryption key. Side channel attacks give the adversaries an advantage to obtain the secret information. Therefore, the security of previous cryptographic schemes is compromised under the circumstances. New model must be constructed to capture such attacks.

In order to guarantee the security of cryptographic systems under some circumstances, we usually define an attack model to limit the attacker's behavior. If the attacker satisfies the constraints, the corresponding cryptosystems are regarded as security in the model. Leakage resilient cryptography is to capture side channel attacks. In fact, it has become a research hotspot in recent years.

For identity-based cryptosystems and traditional public key cryptosystems, some leakage-resilient schemes have been constructed. For certificate-based cryptosystems, as far as we know, no leakage-resilient scheme is presented. The paper puts forward the first certificate-based encryption scheme resilient to master secret key leakage and decryption key leakage.

1.1. Related work

In 2004, Micali and Reyzin (2004) proposed "only computation leaks information" model: computation is divided into many steps. Only the part of the secret state which is accessed (i.e. active) in that step can leak. The other part of the secret state that is not accessed (i.e. inactive) will not leak in that step. Under this model, the leakage-resilient stream cipher (Pietrzak, 2009; Dziembowski and Pietrzak, 2008) and leakage-resilient signature (Faust et al., 2010) were constructed. Although "only computation leaks information" model describes a large class of leakage attacks, it has shortcomings, namely, it does not capture the setting where the inactive part in memory also leaks information (for example, the cold boot attack (Halderman et al., 2009)). In order to solve this problem, the work (Akavia et al., 2009) introduced "bounded leakage" model, and it is a stronger model than "only computation leaks information" model. In "bounded leakage" model, the leakage of inactive part is also considered. Under the "bounded leakage" model, leakage-resilient

* Corresponding author. Tel.: +86 13003416196.
E-mail address: ljg1688@163.com, lijiguo@hhu.edu.cn (J. Li).

http://dx.doi.org/10.1016/j.jss.2015.05.066
0164-1212/

encryption and signature schemes (Chow et al., 2010; Naor and Segev, 2012; Katz and Vaikuntanathan, 2009) were constructed. The constructions of leakage-resilient identity-based schemes attract more attention. Some achievements have been given in the works (Alwen et al., 2009; Chen et al., 2011; Luo et al., 2010).

By constructing the hash proof system, the work (Naor and Segev, 2012) gave the leakage-resilient encryption scheme which can resist $l/4$ bits information leakage about private key (l is the bit length of private key). The work (Alwen et al., 2010) extended the method of the work (Naor and Segev, 2012) to construct the identity-based hash proof system and further to put forward the leakage-resilient identity-based encryption (LR-IBE) in the bounded retrieval model. To improve the property of leakage resilience, the work (Lewko et al., 2011) introduced the dual system encryption.

1.2. Our contribution

Similar to traditional security model of CBE, we consider two types of adversaries as well. The first type of adversary $\mathcal{A}1$ is the malicious user who is allowed to replace public key without knowing the master secret key. The second type of adversary $\mathcal{A}2$ is the dishonest certificate authority (CA) who has the master secret key for generating the certificate but it is not allowed to replace the public key. Inspired by the leakage-resilient certificateless encryption (CLE) (Xiong et al., 2013) and the certificate-based encryption (Wu et al., 2012), we propose the formal definition and the security model of the leakage-resilient certificate-based encryption (LR-CBE) and further present the first leakage-resilient certificate-based encryption scheme in the “bounded leakage” model. The security of the scheme has been proved by utilizing dual system encryption technique. The leakage bound amounts to 1/3 if n is large enough. Performance comparison illustrates the encryption operation of our scheme is faster than that of the schemes given in Gentry (2003). However, decryption cost is linearly correlated with the vector size n. In order to make the scheme more efficient, we can take $n = 2$ and the decryption operation needs 4 pairings which is acceptable in practical application.

1.3. Our technique

In the security proof we use dual system encryption technique proposed in Waters (2009). The dual system encryption technique can be used to improve the security of cryptographic systems. In the dual system encryption the decryption keys and ciphertexts have two states: normal and semi-functional (SF). The normal decryption keys can decrypt the normal and semi-functional ciphertexts. The semi-functional decryption keys can only decrypt the normal ciphertexts correctly. In real security game, all decryption keys and ciphertexts are normal. The security proof is a hybrid argument where the ciphertexts are first altered to semi-functional ones, then, the keys are altered to semi-functional ones gradually. For the consecutive two games we prove that the attacker cannot detect the difference between them with non-negligible advantage. Finally, we give such a game: we only need to produce semi-functional decryption keys and ciphertexts. Thus the attacker cannot correctly decrypt. This allows us to prove security.

1.4. Organization

In Section 2, we give some preliminaries that will be used. Formal description and security model of LR-CBE are given in Section 3. In Section 4, concrete construction of LR-CBE is put forward. Security proof of the proposed scheme is shown in Section 5. The leakage bound is analyzed in Section 6. The comparisons with other schemes are given in Section 7. Section 8 concludes this paper.

2. Preliminaries

2.1. Several basic conceptions

Definition 1. Bilinear Map

Let G and G_T be multiplicative cyclic groups of order q and P be a generator of G, a bilinear map e: $G \times G \to G_T$ has three properties as follows:

(1) Bilinearity: For $P, Q \in G$ and $a, b \in Z*$, $e(P^a, Q^b) = e(P, Q)^{ab}$.
(2) Non-degeneracy: $e(P, P) \neq 1$.
(3) Computability: There is an effective algorithm to calculate $e(P, Q) \in G_T$.

Definition 2. NIZK Proof System

Let R be a binary relation in a language$L$. For $(x, w) \in R$, x is called the statement and w is called the witness. A non-interactive zero-knowledge (NIZK) proof system consists of three algorithms (*Gen, Prf, Ver*). The algorithm *Gen* takes as input a security parameter 1^ϑ and outputs the common reference string *crs*. The prover *Prf* takes as input (*crs, x, w*) and gives an argument or proof π if $(x, w) \in R$. The verifier *Ver* takes as input (*crs*, x, π) and outputs “accept” or “reject”. We call (*Gen, Prf, Ver*) an NIZK proof system for the relation R if it has three properties: soundness, completeness and zero knowledge (Groth, 2010).

Definition 3. Collision-Resistant Hash Function

For the hash function $\overline{H} : \{0, 1\}^* \to \{0, 1\}^k$, the algorithm A can obtain the advantage ε in breaking the collision-resistance of $\overline{H}$ if $\Pr[A(\overline{H}) = (m_0, m_1) : m_0 \neq m_1, \overline{H}(m_0) = \overline{H}(m_1)] \geq \varepsilon$, where the advantage is over the random bits of A. A hash function is collision-resistant if the advantage that any probabilistic polynomial-time (PPT) adversary can obtain is negligible.

2.2. Complexity assumptions

2.2.1. Composite order bilinear groups

Composite order bilinear groups are first introduced in Boneh et al. (2005). Let ψ denote a generator algorithm of composite order bilinear groups. ψ takes as input a security parameter and outputs a description of composite order bilinear groups $\Omega = \{N = p_1 p_2 p_3, G, G_T, e\}$, where p_1, p_2, p_3 are three λ-bit primes (the λ is related to the security parameter and has an influence on the leakage bound which will be analyzed in Section 6), G and G_T are cyclic groups of order $N = p_1 p_2 p_3$ and e is a bilinear map: $G \times G \to G_T$.

Denote G_{p_1}, G_{p_2} and G_{p_3} as the subgroups of G with order p_1, p_2, p_3 respectively. If $h_i \in G_{p_i}$, $h_j \in G_{p_j}$ and $i \neq j$, we have $e(h_i, h_j) = 1$. For example, suppose $h_1 \in G_{p_1}$, $h_2 \in G_{p_2}$, and g is a generator of G. Thus, $g^{p_1 p_2}$ is a generator of G_{p_3}, $g^{p_1 p_3}$ is a generator of G_{p_2} and $g^{p_2 p_3}$ is a generator of G_{p_1}. So, there exists α_1, α_2 such that $h_1 = (g^{p_2 p_3})^{\alpha_1}$ and $h_2 = (g^{p_1 p_3})^{\alpha_2}$. Then, $e(h_1, h_2) = e(g^{p_2 p_3 \alpha_1}, g^{p_1 p_3 \alpha_2}) = e(g^{\alpha_1}, g^{p_3 \alpha_2})^{p_1 p_2 p_3} = 1$. Therefore, G_{p_1}, G_{p_2} and G_{p_3} are mutual orthogonal.

If an element X can be written uniquely as the product of an element of $G_{p_1}$and an element of G_{p_2}, we call them “G_{p_1} part of X” and “G_{p_2} part of X” respectively.

We denote vectors by angle brackets $\langle \cdot, \cdot, \cdot \rangle$ and denote collections of elements of different types by parentheses $(\cdot, \cdot, \cdot)$. Denote dot product of vectors by $\cdot$ and denote component-wise multiplication by $*$. We denote the size or number of bits of the term W as $|W|$.

We define the exponentiation for vectors as follows: For$g \in G$, $\vec{u} = \langle u_1, u_2, \ldots, u_n \rangle \in G^n$, $a \in Z_N$, $\vec{b} = \langle b_1, b_2, \ldots, b_n \rangle \in Z_N$, we define $g^{\vec{b}} = \langle g^{b_1}, g^{b_2}, \ldots, g^{b_n} \rangle$, $\vec{u}^a = \langle {u_1}^a, {u_2}^a, \ldots, {u_n}^a \rangle$. The resulting terms are elements of G^n. For a bilinear group G, we define the pairing operation in G^n: For $\vec{u} = \langle u_1, u_2, \ldots, u_n \rangle \in G^n$ and $\vec{v} = \langle v_1, v_2, \ldots, v_n \rangle \in G^n$, the pairing is $e(\vec{u}, \vec{v}) = \prod_{i=1}^{n} e(u_i, v_i) \in G_T$.

2.2.2. Three assumptions

Here we review three assumptions given in Lewko and Waters (2010), Xiong et al. (2013), Waters (2009), Lewko and Waters (2010) which will be used in our security proof. Denote $G_{p_1 p_2}$ as the subgroup of G with order$p_1 p_2$.

Let ψ be a generator algorithm of composite order bilinear groups. On input a security parameter 1^{ϑ}, ψ outputs a description of composite order bilinear groups. That is, $(N, G, G_T, e) \xleftarrow{R} \psi$, where $N = p_1 p_2 p_3$. Let g_1, g_2 and g_3 be the generators of G_{p_1},$G_{p_2}$and G_{p_3}, respectively.

Assumption 1. Given $D^1 = (N, G, G_T, e, g_1, g_3)$, no PPT adversary succeeds in distinguishing $T_0^1 = g_1^z$ from $T_1^1 = g_1^z g_2^v$ with non-negligible advantage, where $z, v \in Z_N$.

The advantage that adversary $\mathcal{A}$ breaks Assumption 1 is defined as:

$$Adv1_{\psi,\mathcal{A}}(\vartheta) = \ |\Pr[\mathcal{A}(D^1, T_0^1) = 1] - \Pr[\mathcal{A}(D^1, T_1^1) = 1]|.$$

We say that Assumption 1 holds if the advantage $Adv1_{\psi,\mathcal{A}}(\vartheta)$ is negligible for any PPT adversary.

Assumption 2. Given $D^2 = (N, G, G_T, e, g_1, g_3, g_1^z g_2^v, g_2^u g_3^\rho)$ where $z, v, u, \rho \in Z_N$, no PPT adversary succeeds in distinguishing $T_0^2 = g_1^\omega g_3^\sigma$ from $T_1^2 = g_1^\omega g_2^\kappa g_3^\sigma$ with non-negligible advantage, where $\omega, \kappa, \sigma \in Z_N$.

The advantage that adversary $\mathcal{A}$ breaks Assumption 2 is defined as:

$$Adv2_{\psi,\mathcal{A}}(\vartheta) = |\Pr[\mathcal{A}(D^2, T_0^2) = 1] - \Pr[\mathcal{A}(D^2, T_1^2) = 1]|.$$

We say that Assumption 2 holds if the advantage $Adv2_{\psi,\mathcal{A}}(\vartheta)$ is negligible for any PPT adversary.

Assumption 3. Given $D^3 = (N, G, G_T, e, g_1, g_2, g_3, g_1^\alpha g_2^v, g_1^s g_2^u)$ where $\alpha, s, v, u \in Z_N$, no PPT adversary succeeds in distinguishing $T_0^3 = g_1^{\alpha z}$ from $T_1^3 \in G_{p_1}$ with non-negligible advantage.

The advantage that adversary $\mathcal{A}$ breaks Assumption 3 is defined as:

$$Adv3_{\psi,\mathcal{A}}(\vartheta) = \ |\Pr[\mathcal{A}(D^3, T_0^3) = 1] - \Pr[\mathcal{A}(D^3, T_1^3) = 1]|.$$

We say that Assumption 3 holds if the advantage $Adv3_{\psi,\mathcal{A}}(\vartheta)$ is negligible for any PPT adversary.

3. Formal definition and security model of LR-CBE

3.1. Formal definition of LR-CBE

Inspired by the works (Lewko et al., 2011; Xiong et al., 2013), we put forward the formal definition of LR-CBE which is resilient to master secret key leakage and decryption key leakage. We will use a hash function: $\overline{H} : \mathcal{ID} \times \mathcal{PK} \rightarrow \mathcal{ID}$, where $\mathcal{ID}$ is the identity space and $\mathcal{PK}$ is the public key space. The functionality of the hash function is to maintain the security when a CLE is converted to a CBE (refer to Wu et al., 2012). Our LR-CBE scheme is composed of the following seven algorithms.

Setup: *Setup*$(1^{\vartheta}) \rightarrow (mpk, msk)$. The algorithm is run by the **CA**. By taking a security parameter 1^{ϑ} as input, the algorithm generates the master public key *mpk* and the master secret key *msk*. The *mpk* is public to all users. The *mpk* includes the information presentation of the identity space.

SetPrivateKey: *SetPrivateKey*$(ID, mpk) \rightarrow sk_{ID}$. The algorithm is run by the user. It takes as input the master public key *mpk* and the identity *ID*. It outputs the user's private key sk_{ID}.

SetPublicKey: *SetPublicKey*$(ID, sk_{ID}, mpk) \rightarrow pk_{ID}$. The algorithm is run by the user *ID*. It takes as input the master public key *mpk, the identity ID* and the private key sk_{ID}. It outputs the user's public key pk_{ID}.

SetCertificate: *SetCertificate*$(ID, pk_{ID}, mpk, msk) \rightarrow Cert_{ID}$. The algorithm is run by **CA**. For an identity *ID*, it first calculates $\overline{H}(ID, pk_{ID}) \rightarrow ID'$. Then, it takes as input the master public key *mpk*, the master secret key *msk, ID*' and the public pk_{ID}. It outputs the user's certificate $Cert_{ID}$.

Encrypt: *Encrypt*$(ID, mpk, M, pk_{ID}) \rightarrow C$. The algorithm is run by the sender. It takes as input the master public key *mpk*, the plaintext *M*, the receiver's identity *ID* and its corresponding public key pk_{ID}. It outputs the ciphertext *C*.

SetDecryptKey: *SetDecryptKey*$(sk_{ID}, Cert_{ID}) \rightarrow dk_{ID}$. The algorithm generates the decryption key dk_{ID} by using sk_{ID} and $Cert_{ID}$.

Decrypt: *Decrypt*$(mpk, C, dk_{ID}) \rightarrow M$. The algorithm is run by the receiver. It takes as input the master public key *mpk*, the ciphertext *C*, and the decryption key dk_{ID} which is generated by using sk_{ID} and $Cert_{ID}$. It outputs the plaintext *M*.

3.2. Adversaries and oracles of LR-CBE

We consider two types of adversaries in our model like the works (Li et al., 2010, 2012a, 2012b, 2012c, 2013; Lu and Li, 2010, 2012). One type of adversaries is denoted by $\mathcal{A}1$. Another type of adversaries is denoted by $\mathcal{A}2$. $\mathcal{A}1$ acts as the dishonest users. $\mathcal{A}1$ cannot query the certificate of the target user but he is allowed to replace any user's public key. $\mathcal{A}2$ acts as the **CA**. $\mathcal{A}2$ cannot replace the public key of the target user but he may generate any user's certificate.

The security of LR-CBE against $\mathcal{A}1$ is defined by the game $\mathcal{T}1\_Game\_R$ which will be given in the next subsection. In the game the challenger holds a list: $\mathcal{L}1 = (H, ID, ID', pk_{ID}, sk_{ID}, Cert_{ID}, dk_{ID}, l_{dk}, l_{msk})$. The item consists of a handle counter, an identity, a hashed identity, a public key, a private key, a certificate, a decryption key, the amount of decryption key leakage and the amount of master secret key leakage. When an attacker makes a create query $\mathcal{O}-Create$ (refer to the concrete definition in the following), the challenger generates a unique handle *H* and a related item (*H, ID, ID*', pk_{ID}, sk_{ID}, ⊥, ⊥, 0, 0). Given a handle, an adversary can make leakage query. After the leakage query, the value of l_{dk} or l_{msk} which is initially zero will be updated. The other oracle queries will refer the handle.

The security of LR-CBE against $\mathcal{A}2$ is defined by the game $\mathcal{T}2\_Game\_R$ which will be given in the next subsection. In the game the challenger holds a list: $\mathcal{L}2 = (H, ID, ID', pk_{ID}, sk_{ID}, Cert_{ID}, dk_{ID}, l_{dk})$. The item consists of a handle counter, an identity, a hashed identity, a public key, a private key, a certificate, a decryption key and the amount of decryption key leakage. When an attacker makes a create query, the challenger generates a unique handle *H* and a related item (*H, ID, ID*', pk_{ID}, sk_{ID}, ⊥, ⊥, 0). Given a handle, an adversary can make leakage query. After the leakage query, the value of l_{dk} which is initially zero will be updated. The other oracle queries will refer the handle.

We give the oracles that will be used.

(1) $\mathcal{O}-Create$: On the given identity *ID* from the adversary, the challenger $\mathcal{C}$ does as follows. *SetPrivateKey*$(ID, mpk) \rightarrow sk_{ID}$, *SetPublicKey*$(ID, sk_{ID}, mpk) \rightarrow pk_{ID}$. For the adversary $\mathcal{A}1$, it adds an item (*H, ID, ID*', pk_{ID}, sk_{ID}, ⊥, ⊥, 0, 0) in $\mathcal{L}1$. For adversary $\mathcal{A}2$, it calculates: $ID' \leftarrow \overline{H}(ID, pk_{ID})$, *SetCertificate*(*ID*', pk_{ID}, *mpk, msk*) $\rightarrow Cert_{ID}$. It adds an item (*H, ID, ID*', pk_{ID}, sk_{ID}, $Cert_{ID}$, ⊥, 0) in $\mathcal{L}2$. For the two cases, $\mathcal{C}$ will update $H \leftarrow H + 1$. We suppose that the other oracles defined later only respond to the identity which has been created.

(2) $\mathcal{O}-Publickey$: For a handle *H*, the challenger looks up the identity *ID* in the list $\mathcal{L}1$ or $\mathcal{L}2$ and returns the public key pk_{ID} to the adversary $\mathcal{A}1$ or $\mathcal{A}2$.

(3) $O-Replace publickey$: The adversary $\mathcal{A}1$ can replace the public key pk_{ID} of the identity *ID* with a new public key pk'_{ID} of its choice. In order to ensure that the public key pk'_{ID} is valid,

the challenger runs *SetPrivateKey*(*ID, mpk*) $\rightarrow sk_{ID}$ and adds an item (*H, ID, ID*', pk'_{ID}, ⊥, ⊥, ⊥, 0, 0) in the list $\mathcal{L}1$. It updates $H \leftarrow H+1$. The constraint is that for the challenge identity *ID*∗ the adversary $\mathcal{A}1$ is not allowed to replace the public key before the challenge phase and at the same time queries the certificate at some point. Thus $\mathcal{A}1$ obtains a challenge ciphertext about a public key on which it calculates the decryption key.

(4) $\mathcal{O}-Certificate$: For a handle *H*, the challenger looks up the identity *ID* in the list $\mathcal{L}1$. The challenger calculates: $\overline{H}(ID, pk_{ID}) \rightarrow ID'$, *SetCertificate*(*ID*', pk_{ID}, *mpk, msk*) $\rightarrow Cert_{ID}$. It returns the certificate $Cert_{ID}$ to the adversary $\mathcal{A}1$ and updates the item (*H, ID, ID*', pk_{ID}, sk_{ID}, ⊥, ⊥, 0, 0) with (*H, ID, ID*', pk_{ID}, sk_{ID}, $Cert_{ID}$, ⊥, 0, 0).

(5) $\mathcal{O}-Decryptionkey$: The query can be done for the identity that his public key is not replaced. If an adversary queries the decryption key for the identity *ID*, the challenger $\mathcal{C}$ scans the list $\mathcal{L}1$ or $\mathcal{L}2$ to find sk_{ID} and $Cert_{ID}$. Then it calculates the decryption key: $SetDecryptKey(sk_{ID}, Cert_{ID}) \rightarrow dk_{ID}$. For $\mathcal{A}1$, $\mathcal{C}$ outputs dk_{ID} to $\mathcal{A}1$ and updates the item (*H, ID, ID*', pk_{ID}, sk_{ID}, $Cert_{ID}$, ⊥, 0, 0) with (*H, ID, ID*', pk_{ID}, sk_{ID}, $$Cert_{ID}$, dk_{ID}, 0, 0). For $\mathcal{A}2$, $\mathcal{C}$ outputs dk_{ID} to it and updates the item (*H, ID, ID*', pk_{ID}, sk_{ID}, $Cert_{ID}$, ⊥, 0) with (*H, ID, ID*', pk_{ID}, $$sk_{ID}$, $Cert_{ID}$, dk_{ID}, 0).

(6) $\mathcal{O}-Decrypt$: If the adversary queries a decryption for (*ID, C*), the challenger scans the list $\mathcal{L}1$ or $\mathcal{L}2$ to get the decryption key dk_{ID}. The challenger invokes the algorithm **Decrypt** to obtain the corresponding plaintext *M* and gives it to the adversary $\mathcal{A}1$ or $\mathcal{A}2$.

(7) $\mathcal{O}-Leakdecryptionkey$: Given a handle *H* and a leakage function *f* with the output of constant size, the challenger $\mathcal{C}$ looks up the item (*H, ID, ID*', pk_{ID}, sk_{ID}, $Cert_{ID}$, dk_{ID}, l_{dk}, l_{msk})or (*H, ID, ID*', pk_{ID}, sk_{ID}, $Cert_{ID}$, dk_{ID}, l_{dk}) which includes *H* in list $\mathcal{L}1$ or $\mathcal{L}2$. l_{dk} and l_{msk} are initially zero. $\mathcal{C}$ judges if $l_{dk} + |f(dk_{ID})| \le \lambda_{dk}$ where λ_{dk} is the leakage bound of the decryption key. If this is true, the challenger returns the output of $f(dk_{ID})$ to $\mathcal{A}1$ or $\mathcal{A}2$ and updates l_{dk} with $l_{dk} + |f(dk_{ID})|$ in list $\mathcal{L}1$ or $\mathcal{L}2$.

(8) $\mathcal{O}-Leakmasterkey$: Given a handle *H* and a leakage function *f* with the output of constant size, the challenger looks up the item (*H, ID, ID*', pk_{ID}, sk_{ID}, $Cert_{ID}$, dk_{ID}, l_{dk}, l_{msk}) which includes *H* in list $\mathcal{L}1$. It judges if $l_{msk} + |f(msk)| \le \lambda_{msk}$ where λ_{msk} is the leakage bound of the master secret key. If this is true, the challenger returns the output of *f*(*msk*) to $\mathcal{A}1$ and updates l_{msk} with $l_{msk} + |f(msk)|$ in list $\mathcal{L}1$.

3.3. Security model of LR-CBE

In our model there are two types adversaries, so the security is obtained from two security games. The challenger plays the games with the adversary $\mathcal{A}1$ or $\mathcal{A}2$.

3.3.1. Security against type I adversary

The security of LR-CBE against $\mathcal{A}1$ is defined by the game $\mathcal{T}1\_Game\_R$ which is played between the challenger and the adversary $\mathcal{A}1$. $\mathcal{T}1\_Game\_R$ is defined as follows.

$\mathcal{T}1\_Game\_R$:

Initialize: The challenger invokes the algorithm **Setup** to generate the master secret key and the master public key:$Setup(1^{\vartheta}) \rightarrow (mpk, msk)$. The challenger keeps the master secret key as secret and issues the master public key to all users.

Phase 1: The adversary queries the oracles: $\mathcal{O}-Create$, $\mathcal{O}-PublicKey$, $\mathcal{O}-Leakdecryptionkey$, $\mathcal{O}-LeakMasterKey$, $\mathcal{O}-Certificate$, $\mathcal{O}-Replacepublickey$, $\mathcal{O}-Decryptionkey$, $\mathcal{O}-Decrypt$.

The constraints are as follows.

$\mathcal{A}1$ cannot query the decryption key for the challenge identity *ID*∗. For the challenger identity *ID*∗, if $\mathcal{A}1$ replaces the public key, it is not allowed to query the corresponding certificate.

Challenge: The adversary $\mathcal{A}1$ gives two equal length messages $M_0, M_1 \in \mathcal{M}$ and an identity *ID*∗ to the challenger. $\mathcal{M}$ is a given message space. The challenger looks up the corresponding item consisting of the *ID*∗ in list $\mathcal{L}1$. If the item is not in list $\mathcal{L}1$, the challenger will make create query $\mathcal{O}-Create$ for the identity *ID*∗ firstly. Then the challenger randomly selects a bit $\xi \in \{0, 1\}$ and runs algorithm **Encrypt** to generate the ciphertext *C*∗ about the message M_ξ: $Encrypt(ID^*, mpk, M_\xi, pk_{ID^*}) \rightarrow C^*$. At last, the challenger sends the ciphertext *C*∗ to $\mathcal{A}1$.

Phase 2: Just similar to Phase 1, $\mathcal{A}1$ queries the oracles: $\mathcal{O}-Create$, $\mathcal{O}-Publickey$, $\mathcal{O}-Decrypt$, $\mathcal{O}-Replacepublickey$, $\mathcal{O}-Certificate$, $\mathcal{O}-Decryptionkey$. The basic constraints are the same as that of Phase 1. The other constraints are that these oracles cannot query the challenge identity *ID*∗. Furthermore, no leakage query is allowed in this phase. If we allow leakage queries, the adversary can encode the **Decrypt** algorithm of *C*∗ as a leakage function and wins the game trivially.

Guess: At last,$\mathcal{A}1$ guesses a bit ξ' ∈ {0, 1}. If $\xi' = \xi$, $\mathcal{A}1$ wins the game.

The advantage that an adversary $\mathcal{A}1$ wins this game is $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk}, \lambda_{msk}) = |\Pr[\xi' = \xi] - \frac{1}{2}|$.

Definition 4. If there is no PPT adversary $\mathcal{A}1$ who can win $\mathcal{T}1\_Game\_R$ with non-negligible advantage ($Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk}, \lambda_{msk}) \le \varepsilon$ where ε is a negligible value), the LR-CBE is called type I secure against the adaptive chosen ciphertext attacks.

3.3.2. Security against type II adversary

The security of LR-CBE against $\mathcal{A}2$ is defined by the game $\mathcal{T}2\_Game\_R$ which is played between the challenger and the adversary $\mathcal{A}2$. $\mathcal{T}2\_Game\_R$ is defined as follows.

$\mathcal{T}2\_Game\_R$:

Initialize: The challenger invokes the algorithm **Setup** to generate the master public key and the master secret key: $Setup(1^{\vartheta}) \rightarrow (mpk, msk)$. The challenger issues the master public key and the master secret key to $\mathcal{A}2$.

Phase 1: The adversary may query the oracles: $\mathcal{O}-Create$, $\mathcal{O}-Publickey$, $\mathcal{O}-Decryptionkey$, $\mathcal{O}-Decrypt$, $\mathcal{O}-Leakdecryptionkey$. Because $\mathcal{A}2$ knows the master secret key it does not need to query the oracle $\mathcal{O}-Leakmasterkey$ and $\mathcal{O}-Certificate$. The constraints are as follows.

$\mathcal{A}2$ cannot query the decryption key for the challenge identity *ID*∗. $\mathcal{A}2$ is not allowed to replace the public key at any point.

Challenge: The adversary $\mathcal{A}2$ gives two equal length messages $M_0, M_1 \in \mathcal{M}$ and an identity *ID*∗ to the challenger. $\mathcal{M}$ is a given message space. The challenger looks up the corresponding item consisting of the *ID*∗ in list $\mathcal{L}2$. If the item is not in list $\mathcal{L}2$, the challenger will make create query $\mathcal{O}-Create$ for the identity *ID*∗.

Then the challenger randomly selects a bit $\xi \in \{0, 1\}$ and runs algorithm **Encrypt** to get the ciphertext *C*∗ about the message M_ξ: $Encrypt(ID^*, mpk, M_\xi, pk_{ID^*}) \rightarrow C^*$. At last, the challenger sends the ciphertext *C*∗ to the adversary $\mathcal{A}2$.

Phase 2: Just similar to Phase 1, $\mathcal{A}2$ queries the oracles: $\mathcal{O}-Create$, $\mathcal{O}-Publickey$, $\mathcal{O}-Decrypt$, $\mathcal{O}-Decryptionkey$. The basic constraints are the same as that of Phase 1. The other constraints are that these oracles cannot query the challenge identity *ID*∗. Furthermore, no leakage query is allowed in this phase. If we allow leakage queries, the adversary can encode the **Decrypt** algorithm of *C*∗as a leakage function and wins the game trivially.

Guess: At last, $\mathcal{A}2$ guesses a bit ξ' ∈ {0, 1}. If $\xi' = \xi$, $\mathcal{A}2$ wins the game.

The advantage that an adversary $\mathcal{A}2$ wins this game is $Adv_{\mathcal{A}2}^{\mathcal{T}2\_Game\_R}(\lambda_{dk}) = |\Pr[\xi' = \xi] - \frac{1}{2}|$.

Definition 5. If there is no PPT adversary $\mathcal{A}2$ who wins $\mathcal{T}2\_Game\_R$ with non-negligible advantage ($Adv_{\mathcal{A}2}^{\mathcal{T}2\_Game\_R}(\lambda_{dk}) \le \varepsilon$), the LR-CBE is called type II secure against the adaptive chosen ciphertext attacks.

4. Construction of our LR-CBE

We firstly give an NIZK proof system $\prod = (Gen, Prf, Ver)$ which will be employed in our scheme. We define $\prod = (Gen, Prf, Ver)$ is an NIZK proof system with the language $L = \{\beta : Y^{\beta} = Z\}$ where $\beta \in Z_N$, and $Y, Z \in G_T$. $\overline{H} : \mathcal{ID} \times \mathcal{PK} \to \mathcal{ID}$ is a hash function, where $\mathcal{ID}$ is the identity space and $\mathcal{PK}$ is the public key space. The hash function is used to maintain the security when a CLE is converted to a CBE (refer to Wu et al., 2012). Suppose that any identity is an element of Z_N. Our LR-CBE consists of the following seven algorithms.

Setup: It firstly creates composite order bilinear groups $(N = p_1p_2p_3, G, G_T, e)$. Then, it randomly selects $g_1, u_1, h_1, v_1 \in G_{p_1}$ and $g_3 \in G_{p_3}$. It runs algorithm *Gen* of $\prod$ to generate the common reference string *crs* and selects random $(\alpha, x_1, x_2, \dots, x_n, r, y_1, y_2, \dots, y_n) \in Z_N^{2n+2}$ and a vector $\vec{\rho} = \langle \rho_1, \rho_2, \dots, \rho_{n+3}\rangle \in Z_N^{n+3}$ where $n \ge 2$ is an integer. The value of n can be varied. A bigger value of n will generate a bigger leakage rate. Leakage rate is the value that the number of leaked bits from the decryption key or the master secret key divides by the number of bits for the decryption key or master secret key. A smaller value of n will lead to a smaller master public key. It outputs the master public key $mpk = (N, G, G_T, e, e(g_1, v_1)^{\alpha}, g_1, g_1^{x_1}, \dots, g_1^{x_n}, u_1, h_1, v_1, g_3, crs)$ and the master secret key $msk = (\vec{K}, K_1, K_2, K_3) = (\langle v_1^{y_1}, \dots, v_1^{y_n}\rangle, g_1^{\alpha} h_1^{-r} \prod_{i=1}^{n} g_1^{-x_i y_i}, v_1^{r}, u_1^{r}) * g_3^{\vec{\rho}}$.

SetPrivateKey: The user sets the private key $sk_{ID} = \beta$ where $\beta \in Z_N$.

SetPublicKey: The user sets public key $pk_{ID} = (Y, \pi) = (e(g_1, v_1)^{\alpha\beta}, \pi)$ where $\pi \leftarrow Prf(crs, (e(g_1, v_1)^{\alpha\beta}, e(g_1, v_1)^{\alpha}), \beta)$ is an NIZK proof that β is the discrete logarithm of $e(g_1, v_1)^{\alpha\beta}$ to the base $e(g_1, v_1)^{\alpha}$.

SetCertificate: The **CA** randomly selects a vector $\vec{\rho}' = \langle \rho'_1, \rho'_2, \dots, \rho'_{n+2}\rangle \in Z_N^{n+2}$ and $n+1$ elements $(r', z_1, \dots, z_n) \in Z_N^{n+1}$. Then, it calculates the certificate as follows: $\overline{H}(ID, pk_{ID}) = ID'$, $Cert_{ID} = (\vec{D}, D_1, D_2) = (\vec{K}, K_1, K_2) * (\langle v_1^{z_1}, \dots, v_1^{z_n}\rangle, (K_3)^{-ID'} (u_1^{ID'} h_1)^{-r'} \prod_{i=1}^{n} g_1^{-x_i z_i}, v_1^{r'}) * g_3^{\vec{\rho}'}$. The G_{p_1} part of $Cert_{ID}$ can be viewed as $(\langle v_1^{z'_1}, \dots, v_1^{z'_n}\rangle, g_1^{\alpha} (u_1^{ID'} h_1)^{-r''} \prod_{i=1}^{n} g_1^{-x_i z'_i}, v_1^{r''})$ for some $(r'', z'_1, \dots, z'_n) \in Z_N^{n+1}$ where $z'_i = y_i + z_i$for $i = 1$ to n and $r'' = r + r'$.

Encrypt: The sender verifies the validation for proof π. If π is valid, it picks randomly $s \in Z_n$ and computes the ciphertext: $C = (C_0, \vec{C}, C_1, C_2) = (M.e(g_1, v_1)^{\alpha\beta s}, \langle g_1^{x_1 s}, \dots, g_1^{x_n s}\rangle, v_1^{s}, (u_1^{ID'} h_1)^{s})$, where $ID' = \overline{H}(ID, pk_{ID})$.

SetDecryptKey: The user picks $\vec{\rho}'' = \langle \rho''_1, \dots, \rho''_{n+2}\rangle \in Z_N^{n+2}$, $\vec{w} = \langle w_1, \dots, w_n\rangle \in Z_N^{n}$ and $t \in Z_N$ randomly. The user calculates the decryption key as follows: $\overline{H}(ID, pk_{ID}) = ID'$, $dk_{ID} = (\vec{S}, S_1, S_2) = (\vec{D}, D_1, D_2)^{\beta} * (\langle v_1^{w_1}, \dots, v_1^{w_n}\rangle, (u_1^{ID'} h_1)^{-t} \prod_{i=1}^{n} g_1^{-x_i w_i}, v_1^{t}) * g_3^{\vec{\rho}''}$. The G_{p_1} part of dk_{ID} can be viewed as $(\langle v_1^{w'_1}, \dots, v_1^{w'_n}\rangle, g_1^{\alpha\beta} (u_1^{ID'} h_1)^{-t'} \prod_{i=1}^{n} g_1^{-x_i w'_i}, v_1^{t'})$ for some $(t', w'_1, \dots, w'_n) \in Z_N^{n+1}$ where $w'_i = w_i + (y_i + z_i)^{\beta}$ for $i = 1$ to n and $t' = t + (r + r')^{\beta}$.

Decrypt: By using the decryption key, the user gets $M = \frac{C_0}{e(\vec{C}, \vec{S}) \cdot e(C_1, S_1) \cdot e(C_2, S_2)}$.

Correctness.

$$e(\vec{C}, \vec{S}) \cdot e(C_1, S_1) \cdot e(C_2, S_2) = e(\langle g_1^{x_1 s}, \dots, g_1^{x_n s}\rangle, \ \langle v_1^{w'_1}, \dots, v_1^{w'_n}\rangle * \langle g_3^{(\rho_1 + \rho'_1)^{\beta} + \rho''_1}, \dots, g_3^{(\rho_n + \rho'_n)^{\beta} + \rho''_n}\rangle).$$

$$e\left(v_1^{s}, g_1^{\alpha\beta} (u_1^{ID'} h_1)^{-t'} \prod_{i=1}^{n} g_1^{-x_i w'_i} . g_3^{\rho''_{n+1}}\right) \cdot e((u_1^{ID'} h_1)^{s}, v_1^{t'} \cdot g_3^{\rho''_{n+2}})$$

$$= e(\langle g_1^{x_1 s}, \dots, g_1^{x_n s}\rangle, \ \langle v_1^{w'_1} \cdot g_3^{(\rho_1 + \rho'_1)^{\beta} + \rho''_1}, \dots, v_1^{w'_n} \cdot g_3^{(\rho_n + \rho'_n)^{\beta} + \rho''_n}\rangle).$$

$$e\left(v_1^{s}, g_1^{\alpha\beta} (u_1^{ID'} h_1)^{-t'} \prod_{i=1}^{n} g_1^{-x_i w'_i} . g_3^{\rho''_{n+1}}\right) \cdot e((u_1^{ID'} h_1)^{s}, v_1^{t'} \cdot g_3^{\rho''_{n+2}})$$

$$= \prod_{i=1}^{n} e(g_1^{x_i s}, v_1^{w'_i} \cdot g_3^{(\rho_i + \rho'_i)^{\beta} + \rho''_i})$$
$$\cdot e\left(v_1^{s}, g_1^{\alpha\beta} (u_1^{ID'} h_1)^{-t'} \prod_{i=1}^{n} g_1^{-x_i w'_i} . g_3^{\rho''_{n+1}}\right) \cdot e((u_1^{ID'} h_1)^{s}, v_1^{t'} \cdot g_3^{\rho''_{n+2}})$$

$$= \prod_{i=1}^{n} e(g_1^{x_i s}, v_1^{w'_i}) \cdot \prod_{i=1}^{n} e(g_1^{x_i s}, g_3^{(\rho_i + \rho'_i)^{\beta} + \rho''_i})$$
$$\cdot e\left(v_1^{s}, g_1^{\alpha\beta} (u_1^{ID'} h_1)^{-t'} \prod_{i=1}^{n} g_1^{-x_i w'_i}\right) \cdot e(v_1^{s}, g_3^{\rho''_{n+1}}).$$
$$e((u_1^{ID'} h_1)^{s}, v_1^{t'}) \cdot e((u_1^{ID'} h_1)^{s}, g_3^{\rho''_{n+2}})$$

$$= \prod_{i=1}^{n} e(g_1^{x_i s}, v_1^{w'_i}) \cdot e\left(v_1^{s}, g_1^{\alpha\beta} (u_1^{ID'} h_1)^{-t'} \prod_{i=1}^{n} g_1^{-x_i w'_i}\right) \cdot e((u_1^{ID'} h_1)^{s}, v_1^{t'})$$

$$= \prod_{i=1}^{n} e(g_1^{x_i s}, v_1^{w'_i}) \cdot e\left(v_1^{s}, \prod_{i=1}^{n} g_1^{-x_i w'_i}\right) \cdot e(v_1^{s}, g_1^{\alpha\beta}) \cdot e(v_1^{s}, (u_1^{ID'} h_1)^{-t'})$$
$$\cdot e((u_1^{ID'} h_1)^{s}, v_1^{t'})$$

$$= \prod_{i=1}^{n} e(g_1^{x_i w'_i}, v_1^{s}) \cdot e\left(v_1^{s}, \prod_{i=1}^{n} g_1^{-x_i w'_i}\right) \cdot e(v_1, g_1)^{\alpha\beta s} \cdot e(v_1^{t'}, (u_1^{ID'} h_1)^{-s})$$
$$\cdot e((u_1^{ID'} h_1)^{s}, v_1^{t'})$$

$$= e\left(\prod_{i=1}^{n} g_1^{x_i w'_i}, v_1^{s}\right) . e\left(v_1^{s}, \prod_{i=1}^{n} g_1^{-x_i w'_i}\right) \cdot e(v_1, g_1)^{\alpha\beta s}$$

$$= e(v_1, g_1)^{\alpha\beta s}$$

5. Security proof

Inspired by dual system encryption method (Lewko et al., 2011, Waters, 2009, Lewko and Waters, 2010), we use semi-functional ciphertexts and keys in our proof. In order to accomplish our proof, we give dual system construction of our LR-CBE.

5.1. Dual system description of our LR-CBE

DS-Setup: The algorithm is based on **Setup**. It outputs a normal master public key and a semi-functional master secret key $\widetilde{msk}$.

DS-SetCertificate: The algorithm is based on **SetCertificate**. It is run by the certificate authority. It takes as input master public key *mpk*, the SF master secret key $\widetilde{msk}$, the identity *ID* and the corresponding public pk_{ID}. It outputs the user's SF certificate $\widetilde{Cert_{ID}}$.

DS-SetDecryptKey: The algorithm is run by the user ID. It takes as input the master public key. It outputs the user's semi-functional decryption key $\widetilde{dk_{ID}}$.

DS-Encrypt: The algorithm is run by the sender. It takes as input the master public key mpk, the plaintext M, the receiver's identity ID and the corresponding public pk_{ID}. It outputs the SF ciphertext $\widetilde{C}$.

The SF decryption keys can only decrypt normal ciphertexts. The normal decryption keys can decrypt normal and semi-functional ciphertexts. Of course, if the input of the algorithm **SetCertificate** is the SF master secret key $\widetilde{msk}$, the output of **SetCertificate** is SF certificate $\widetilde{Cert_{ID}}$. If the input of the algorithm **SetDecryptKey** is SF certificate $\widetilde{Cert_{ID}}$, the output of the algorithm **SetDecryptKey** is SF decryption key $\widetilde{dk_{ID}}$.

Here, we use $\widetilde{X}$ to denote the semi-functional construction of X. When the semantic context is clear, we also use X to denote the corresponding semi-functional construction.

5.2. Dual system construction of our LR-CBE

In Section 4, keys and ciphertexts are normal. That is to say, they do not contain G_{p_2} part. Here, we give the dual system construction of our LR-CBE according to the description in Section 5.1.

DS-Setup: The algorithm invokes **Setup** to generate a normal master secret key $msk = (\vec{K}, K_1, K_2, K_3)$. It selects $\vec{\iota} \in Z_N^n$, $(\iota_1, \iota_2, \iota_3) \in Z_N^3$ randomly, then computes the semi-functional master secret key $\widetilde{msk} = (\vec{K} * g_2^{\vec{\iota}}, K_1 \cdot g_2^{\iota_1}, K_2 \cdot g_2^{\iota_2}, K_3 \cdot g_2^{\iota_3})$.

DS-SetCertificate: The algorithm runs **SetCertificate** to generate the normal certificate $Cert_{ID} = (\vec{D}, D_1, D_2)$. It selects $\vec{\eta} \in Z_N^n$, $(\eta_1, \eta_2) \in Z_N^2$ randomly, then computes the semi-functional certificate $\widetilde{Cert_{ID}} = (\vec{D} * g_2^{\vec{\eta}}, D_1 \cdot g_2^{\eta_1}, D_2 \cdot g_2^{\eta_2})$.

DS-SetDecryptKey: The algorithm runs **SetDecryptKey** to generate the normal decryption key $dk_{ID} = (\vec{S}, S_1, S_2)$. Next, it randomly selects $\vec{\theta} \in Z_N^n$, $(\theta_1, \theta_2) \in Z_N^2$ and computes the semi functional decryption key $\widetilde{dk_{ID}} = (\vec{S} * g_2^{\vec{\theta}}, S_1 \cdot g_2^{\theta_1}, S_2 \cdot g_2^{\theta_2})$.

DS-Encrypt: The algorithm invokes **Encrypt** to get normal ciphertext $C = (C_0, \vec{C}, C_1, C_2)$. Next, it selects $\vec{\delta} \in Z_N^n$, $(\delta_1, \delta_2) \in Z_N^2$ randomly and computes the semi-functional ciphertext $\widetilde{C} = (C_0, \vec{C} * g_2^{\vec{\delta}}, C_1 \cdot g_2^{\delta_1}, C_2 \cdot g_2^{\delta_2})$.

If $\vec{\theta} \cdot \vec{\delta} + \theta_1\delta_1 + \theta_2\delta_2 = 0 \mod p_2$, the SF decryption key is called as nominal semi-functional (NSF) decryption key. In this case, even if the ciphertext is semi functional, **Decrypt** will be done successfully. Otherwise, we call the SF decryption key as true SF decryption key.

5.3. Security of our LR-CBE

5.3.1. Security proof against type I adversary

In order to prove the security with leakage resilience, we give a series of games here, which are modifications of the game $\mathcal{T}1\_Game\_R$. If we prove that these games are indistinguishable we finish the security proof of our scheme. We use Q to denote the maximum number of $\mathcal{O}-Create$ queries in the games.

These additional games are as follows.

$\mathcal{T}1\_Game\_0$: It is nearly the same as $\mathcal{T}1\_Game\_R$ besides the challenge ciphertext. In $\mathcal{T}1\_Game\_0$ the challenge ciphertext is semi-functional.

$\mathcal{T}1\_Game\_k$ ($k = 1$ to Q): The challenge ciphertext is semi-functional. For the adversary's queries, the challenger answers in two ways. Toward the first k queries, the challenger does as follows.

If the adversary makes a certificate query or decryption key query, the challenger creates the SF certificate and SF decryption key. If the adversary makes a replace public key query, the challenger only creates the SF certificate. The challenger adds the corresponding item in list $\mathcal{L}1$ and gives it to the adversary.

For the remaining queries, the challenger creates normal certificate and normal decryption key. $\mathcal{T}1\_Game\_Msk$: It is nearly the same as $\mathcal{T}1\_Game\_Q$ except that in $\mathcal{T}1_Game_Msk$the master secret key is semi-functional. Thus, for $\mathcal{O}-Leakmasterkey$ query, the challenger creates semi-functional master secret key and sends the output of the leakage function to the adversary. The leakage function takes the semi-functional master secret key as input.

Table 1
The types of master secret keys, ciphertexts and decryption keys in different games.

Games	The type of master secret keys, ciphertexts and decryption keys: $(T_M, T_C, (T_D, \ldots, T_D))$
$\mathcal{T}1\_Game\_R$	$(N, N, (N, \ldots, N))$
$\mathcal{T}1\_Game\_0$	$(N, SF, (N, \ldots, N))$
$\mathcal{T}1\_Game\_k$ $k \in (1, \ldots, Q-1)$	$(N, SF, (SF, \ldots, \underset{k}{SF}, N, \ldots, N))$
$\mathcal{T}1\_Game\_Q$	$(N, SF, (SF, \ldots, SF))$
$\mathcal{T}1\_Game\_Msk$	$(SF, SF, (SF, \ldots, SF))$
$\mathcal{T}1\_Game\_Final$	$(SF, SF, (SF, \ldots, SF))$

$\mathcal{T}1\_Game\_Final$: It is nearly the same as $\mathcal{T}1\_Game\_Msk$ except that the challenger randomly selects a message M_ς and encrypts it. In $\mathcal{T}1\_Game\_Msk$, the challenger encrypts a challenge message M_ξ. From the adversary's point of view, in $\mathcal{T}1\_Game\_Final$ the bit ξ' of his choice is independent of the challenger's bit ξ.

In Table 1, we explain the types of master secret key, ciphertexts and decryption keys which are created in different games. Let SF denote the semi-functional key or ciphertext. Let N denote the normal key or ciphertext. We use T_M, T_C and T_D to denote the types of master secret key, ciphertext and decryption key, respectively. In each game, the maximum number of create query $\mathcal{O}-Create$ is Q. Thus, we use $(\underbrace{(T_M, T_C, T_D), \ldots, (T_M, T_C, T_D)}_{Q})$ to denote the according types of Q create queries in a game. For every game above, the types of the ciphertexts are the same in every create query. At the same time, the types of master keys are the same in every create query. Thus, $((T_M, T_C, T_D), \ldots, (T_M, T_C, T_D))$ could be shortened as $(T_M, T_C, (\underbrace{T_D, \ldots T_D}_{Q}))$ to denote the according types of Q create queries in a game.

Theorem 1. *Under Assumptions 1–3, for $\lambda_{dk} = (n - 2c - 1)\lambda$ bits leakage of the decryption key and $\lambda_{msk} = (n - 2c - 1)\lambda$ bits leakage of the master secret key, the LR-CBE scheme is secure against $\mathcal{A}1$ where $n \geq 2$ is an integer and c is a fixed positive constant.*

The value of n can be varied. A larger value of n will generate a larger leakage rate. The smaller value of n will lead to a smaller master public key. The concrete explanation is given in Section 6.

Proof. The general idea of proof is as follows. We will use a series of games $\mathcal{T}1\_Game\_R$, $\mathcal{T}1\_Game\_k$ ($k \in (0, 1, \ldots, Q)$), $\mathcal{T}1\_Game\_Msk$ and $\mathcal{T}1\_Game\_Final$ and Lemmas 1–9 to finish our security proof. On the one hand, we prove that these games are indistinguishable by Lemmas 2–9. On the other hand, the advantage that the adversary wins in $\mathcal{T}1\_Game\_Final$ is negligible. We get the leakage bound by Lemma 1. Thus, we can obtain the security of the scheme.

A bit more precisely, we use Table 2 to show the difference of advantages that the adversary wins in the successive two games. Thus, it is easy to obtain the security by us. Here, we only use the results of Lemmas 1–9. The specific proofs of Lemmas 1–9 are given in the following. Let $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk}, \lambda_{msk})$ denote the advantage that the adversary $\mathcal{A}1$ wins in game $\mathcal{T}1\_Game\_k$ ($k \in (1, \ldots, Q)$). Let $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Msk}(\lambda_{dk}, \lambda_{msk})$ denote the advantage that the adversary $\mathcal{A}1$ wins in game $\mathcal{T}1\_Game\_Msk$. Let $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk}, \lambda_{msk})$ denote the advantage that the adversary $\mathcal{A}1$ wins in game$\mathcal{T}1_Game_Final$.

Table 2
The difference of advantages that the adversary wins in the successive two games.

Two successive games	Difference of advantages	Related lemmas
$\mathcal{T}1\_Game\_R$ and $\mathcal{T}1\_Game\_0$	$\lvert Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_0}(\lambda_{dk},\lambda_{msk})\rvert \le \varepsilon$	Lemma 2
$\mathcal{T}1\_Game\_k$ and $\mathcal{T}1\_Game\_k+1$ $k\in(0,1,\dots,Q-1)$	$\lvert Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k+1}(\lambda_{dk},\lambda_{msk})\rvert \le \varepsilon$	Lemma 3 Lemma 4 Lemma 5 Lemma 6
$\mathcal{T}1\_Game\_Q$ and $\mathcal{T}1\_Game\_Msk$	$\lvert Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Q}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Msk}(\lambda_{dk},\lambda_{msk})\rvert \le \varepsilon$	Lemma 7 Lemma 8
$\mathcal{T}1\_Game\_Msk$ and $\mathcal{T}1\_Game\_Final$	$\lvert Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Q}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk},\lambda_{msk})\rvert \le \varepsilon$	Lemma 9

From Table 2, the security of our scheme is obtained directly. We have:

$$\left|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk},\lambda_{msk})\right|$$

$$\begin{aligned}
&|Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_R}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_0}(\lambda_{dk},\lambda_{msk})\\
&+Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_0}(\lambda_{dk},\lambda_{msk}) - \dots\\
&= -Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_k}(\lambda_{dk},\lambda_{msk}) + Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_k}(\lambda_{dk},\lambda_{msk}) - \dots\\
&- Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_Msk}(\lambda_{dk},\lambda_{msk}) + Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_Msk}(\lambda_{dk},\lambda_{msk})\\
&- Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_Final}(\lambda_{dk},\lambda_{msk})|\\
&|Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_R}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_0}(\lambda_{dk},\lambda_{msk})|\\
&+|Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_0}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_1}(\lambda_{dk},\lambda_{msk})| + \dots\\
&\le +|Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_k}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_k+1}(\lambda_{dk},\lambda_{msk})| + \dots\\
&+|Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_Q}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_Msk}(\lambda_{dk},\lambda_{msk})|\\
&+|Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_Msk}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1_Game_Final}(\lambda_{dk},\lambda_{msk})|
\end{aligned}$$

$$\le \varepsilon + (Q+1)\varepsilon + \varepsilon = (Q+3)\varepsilon.$$

So, $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk},\lambda_{msk})| \le (Q+3)\varepsilon$. What is more, $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk},\lambda_{msk}) \le \varepsilon$ is proved similarly to that of Theorem 6.8 in the full version of Lewko et al. (2011). To sum up, $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk},\lambda_{msk}) \le \varepsilon$. In addition, Lemma 1 proves the leakage bound. Thus, we finish the proof of Theorem 1.

The specific proofs of Lemmas 1–9 are given as follows.

Lemma 1. *The leakage size is at most* $\lambda_{dk} = \lambda_{msk} = (n-2c-1)\lambda$.

Proof. We first introduce a useful conclusion (we called it as Conclusion 1) given in Brakerski et al. (2010).

Conclusion 1. Let p be a prime. Let $m \ge l \ge 2$ $(m, l \in N)$. Let $X \leftarrow Z_p^{m\times l}$, $T \leftarrow Rk_1(Z_p^{l\times 1})$, $U \leftarrow Z_p^m$. $Rk_1(Z_p^{l\times 1})$ denotes that the rank of $Z_p^{l\times 1}$ is 1. For an arbitrary leakage function $f: Z_p^m \to W$, if $|W| \le 4\cdot(1-\frac{1}{p})\cdot p^{l-1}\cdot\varepsilon^2$ we have the statistical distance $SD((X, f(X\cdot T)), (X, f(U))\le \varepsilon$ where ε is a negligible value.

In our paper, we use the following Corollary 1.

Corollary 1. *Let p be a prime. Let* $\vec{\delta} \leftarrow Z_p^m$, $\vec{\tau} \leftarrow Z_p^m$, $\vec{\tau}' \leftarrow Z_p^m$ *such that* $\vec{\tau}'$ *is orthogonal to* $\vec{\delta}$ *modulo p under the dot product, where* $m \ge 3$ $(m \in N)$. *For an arbitrary leakage function* $f: Z_p^m \to W$, *if* $|W| \le 4\cdot(1-\frac{1}{p})\cdot p^{n-1}\cdot\varepsilon^2$ *we have* $SD((\vec{\delta}, f(\vec{\tau}')), (\vec{\delta}, f(\vec{\tau}))) \le \varepsilon$ *where* ε *is a negligible value.*

Proof. We apply Conclusion 1 with $l = m-1$. Then, $\vec{\tau}$ corresponds to U and the basis of the orthogonal space of $\vec{\delta}$ corresponds to X. We will see that $\vec{\tau}'$ is distributed as $X\cdot T$ where $T \leftarrow Rk_1(Z_p^{(m-1)\times 1})$. Because $\vec{\delta} \in Z_p^m$ is selected uniformly at random, $X \leftarrow Z_p^{m\times(m-1)}$ is determined by $\vec{\delta}$. Thus, we have $SD((\vec{\delta}, f(\vec{\tau}')), (\vec{\delta}, f(\vec{\tau}))) = SD((X, f(X\cdot T)), (X, f(U))$.

If we let $n+1 = m$, $p_2 = p$ and $\varepsilon = p_2^{-c}$, we get that the leakage size is at most $\log|W| \le (n-1)\log p_2 - 2c\log p_2 = (n-2c-1)\log p_2 = (n-2c-1)\lambda$, where $\log p_2 = \lambda$. Thus, we get that the leakage size is at most $\lambda_{dk} = \lambda_{msk} = (n-2c-1)\,\lambda$.

Lemma 2. *Under Assumption 1, the advantage that PPT adversary* $\mathcal{A}1$ *succeeds in distinguishing* $\mathcal{T}1\_Game\_R$ *and* $\mathcal{T}1\_Game\_0$ *is negligible. That is* $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_0}(\lambda_{dk},\lambda_{msk})| \le \varepsilon$.

Proof. If there is a PPT adversary $\mathcal{A}1$who distinguishes $\mathcal{T}1\_Game\_R$ and $\mathcal{T}1\_Game\_0$ with non-negligible advantage, we can construct a simulator $\mathcal{B}$ to break Assumption 1.

Firstly, $\mathcal{B}$ is given an instance(N, G, G_T, e, g_1, g_3) and a challenge item T, where $T = g_1^z$ or $T = g_1^z g_2^v$ for $z, v \in Z_N$. $\mathcal{B}$ interacts with $\mathcal{A}1$ as follows.

Setup phase: $\mathcal{B}$ firstly creates composite order bilinear groups $(N = p_1p_2p_3, G, G_T, e)$. Then, $\mathcal{B}$ selects $a, b, d \in Z_N$ uniformly at random and sets $u_1 = g_1^a$, $h_1 = g_1^b$ and $v_1 = g_1^d$. Given $g_1 \in G_{p_1}$ and $g_3 \in G_{p_3}$, $\mathcal{B}$ runs the algorithm Gen of $\prod$ to generate the common reference string crs and selects random $(\alpha, x_1, x_2, \dots, x_n, r, y_1, y_2, \dots, y_n) \in Z_N^{2n+2}$ and a vector $\vec{\rho} = \langle\rho_1, \rho_2, \dots, \rho_{n+3}\rangle \in Z_N^{n+3}$ where $n \ge 2$ is an integer.

It outputs the master public key $mpk = (N, G, G_T, e, e(g_1, v_1)^\alpha, g_1, g_1^{x_1}, \dots, g_1^{x_n}, u_1, h_1, v_1, g_3, crs)$ and the master secret key $msk = (\vec{K}, K_1, K_2, K_3) = (\langle v_1^{y_1}, \dots, v_1^{y_n}\rangle, g_1^\alpha h_1^{-r}\prod_{i=1}^{n} g_1^{-x_iy_i}, v_1^r, u_1^r) * g_3^{\vec{\rho}}$.

Oracle query: For the public key which is not replaced, $\mathcal{B}$ knows the decryption key because it has the msk. Hence, $\mathcal{B}$ can answer the adversary's queries. In addition, $\mathcal{B}$ stores the private key of each user.

Challenge: $\mathcal{A}1$ gives challenge identity $ID*$ and two message M_0 and M_1 to $\mathcal{B}$. $\mathcal{B}$ scans list $\mathcal{L}1$ to find public key (Y, π) of $ID*$ where $Y = e(g_1, v_1)^{\alpha\beta}$. If the $\mathcal{A}1$ does not replace the public key $\mathcal{B}$ knows β. Otherwise, $\mathcal{B}$ extracts β from π by NIZK proof. $\mathcal{B}$ selects $\xi \in \{0, 1\}$ at random and outputs the ciphertext: $C^* = (C_0^*, \vec{C}^*, C_1^*, C_2^*) = (M_\xi\cdot e(v_1^\beta, T), \langle T^{x_1}, \dots, T^{x_n}\rangle, T^d, T^{aID^*+b})$.

When $T = g_1^z$ (that is, no G_{p_2} part is contained in T), the given ciphertext is normal. Thus, $\mathcal{B}$ simulates the $\mathcal{T}1\_Game\_R$ correctly.

When $T = g_1^z g_2^v$, we set $\vec{\delta} = \langle x_1, \dots, x_n\rangle$, $\delta_1 = d$, $\delta_2 = aID^* + b$. We can see that $\vec{\delta}$, δ_1 and δ_2 are distributed uniformly at random from the adversary's view. The reason is that the $x_1, \dots, x_n, d, a$ and b are merely calculated under modulo p_1 in mpk. From the adversary's point of view, $x_1, \dots, x_n, d, a$ and b are random under modulop_2. Thus, $\mathcal{B}$ simulates the $\mathcal{T}1\_Game\_0$ correctly.

If $\mathcal{A}1$ can distinguish $\mathcal{T}1\_Game\_R$ and $\mathcal{T}1\_Game\_0$ with non-negligible advantage, $\mathcal{B}$ may break Assumption 1 by $\mathcal{A}1$ with non-negligible advantage. Namely, $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_0}(\lambda_{dk},\lambda_{msk})| \le \varepsilon$.

Lemma 3. *If* $\lambda_{dk} = (n-2c-1)\,\lambda$ *and Assumption 2 holds, the advantage that PPT adversary* $\mathcal{A}1$ *distinguishes* $\mathcal{T}1\_Game\_k$ *and* $\mathcal{T}1\_Game\_k+1$ *is negligible. That is* $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k+1}(\lambda_{dk},\lambda_{msk})| \le \varepsilon$.

Proof. In order to prove Lemma 3, another game which is called $\mathcal{T}1\_AltGame\_k$ is defined. Compared with $\mathcal{T}1\_Game\_k$, in $\mathcal{T}1\_AltGame\_k$ the certificate is normal instead of semi-functional if the kth query are create oracle. The decryption key is still semi-functional. Therefore, Lemma 3 can be proved by the following three lemmas.

Lemma 4. *If* $\lambda_{dk} = (n-2c-1)\lambda$ *and Assumption 2 holds, the advantage that PPT adversary* $\mathcal{A}1$ *can succeed in distinguishing* $\mathcal{T}1\_Game\_k$ *and* $\mathcal{T}1\_AltGame\_k$ *is negligible. That is* $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk},\lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_AltGame\_k}(\lambda_{dk},\lambda_{msk})| \le \varepsilon$.

Compared with $\mathcal{T}1\_Game\_k$, in $\mathcal{T}1\_AltGame\_k$ the certificate is normal instead of semi-functional if the kth query are create oracle. The decryption key is still semi-functional.

Proof. If there is a PPT adversary $\mathcal{A}1$ who distinguishes $\mathcal{T}1_Game_k$and $\mathcal{T}1\_AltGame\_k$ with non-negligible advantage, we can construct a simulator $\mathcal{B}$ who can break Assumption 2. Firstly, $\mathcal{B}$ is given an instance $(N, G, G_T, e, g_1, g_3, g_1^z g_2^\nu, g_2^\mu g_3^\rho)$ and a challenge item T which is $g_1^\omega g_3^\sigma$ or $g_1^\omega g_2^\kappa g_3^\sigma$. $\mathcal{B}$ interacts with $\mathcal{A}1$ as follows.

Setup phase: $\mathcal{B}$ firstly creates composite order bilinear groups $(N = p_1p_2p_3, G, G_T, e)$. $\mathcal{B}$ selects a, b, $d \in Z_N$ uniformly at random and sets $u_1 = g_1^a$, $h_1 = g_1^b$ and $\nu_1 = g_1^d$. Given $g_1 \in G_{p_1}$ and $g_3 \in G_{p_3}$, $\mathcal{B}$ runs the algorithm Gen of $\prod$ to generate the common reference string crs and selects random $(\alpha, x_1, x_2, \dots, x_n, r, y_1, y_2, \dots, y_n) \in Z_N^{2n+2}$and a vector $\vec{\rho} = \langle \rho_1, \rho_2, \dots, \rho_{n+3}\rangle \in Z_N^{n+3}$ where $n \geq 2$ is an integer.

It outputs the master public key $mpk = (N, G, G_T, e, e(g_1, \nu_1)^\alpha, g_1, g_1^{x_1}, \dots, g_1^{x_n}, u_1, h_1, \nu_1, g_3, crs)$ and the master secret key $msk = (\vec{K}, K_1, K_2, K_3) = (\langle \nu_1^{y_1}, \dots, \nu_1^{y_n}\rangle >, g_1^\alpha h_1^{-r} \prod_{i=1}^{n} g_1^{-x_i y_i}, \nu_1^r, u_1^r) * g_3^{\vec{\rho}}$.

Oracle query: $\mathcal{B}$ may reply every query because it knows α, x_i, y_i, r. Specifically, $\mathcal{B}$ answers the jth query for ID_j as follows.

(1) If it is the create query, $\mathcal{B}$ generates normal certificate $Cert_{ID_j}$ by using msk. The $Cert_{ID_j}$ can leak information. $\mathcal{B}$ runs **SetPrivateKey** to get β and runs **SetPublicKey** to get pk_{ID_j}. $\mathcal{B}$ picks randomly $(\omega_1, \omega_2, \dots, \omega_n, t) \in Z_N^{n+1}$ and $\vec{\rho} \in Z_N^{n+2}$.
  (a) If $j \leq k$, $\mathcal{B}$ selects randomly a vector $\vec{\gamma} \in Z_N^{n+2}$ and calculates the decryption key $dk_{ID_j} = (\langle \nu_1^{\omega_1}, \dots, \nu_1^{\omega_n}\rangle, g^{\alpha\beta}(u_1^{ID_j} h_1)^{-t} \prod_{i=1}^{n} g_1^{-x_i\omega_i}, \nu_1^t) * (g_2^\mu g_3^\rho)^{\vec{\gamma}} * g_3^{\vec{\rho}}$.
  (b) If $j > k+1$, $\mathcal{B}$ calculates the decryption key $dk_{ID_j} = (\langle \nu_1^{\omega_1}, \dots, \nu_1^{\omega_n}\rangle, g^{\alpha\beta}(u_1^{ID_j} h_1)^{-t} \prod_{i=1}^{n} g_1^{-x_i\omega_i}, \nu_1^t) * g_3^{\vec{\rho}}$.
  (c) If $j = k+1$, $\mathcal{B}$ calculates the decryption key $dk_{ID_{k+1}} = (\langle T^{dw_1}, \dots, T^{dw_n}\rangle, T^{-(aID_{k+1}+b)} * \prod_{i=1}^{n} T^{-x_i\omega_i} * g_1^{\alpha\beta}, T^d) * g_3^{\vec{\rho}}$ by using the challenge item T.

If $T = g_1^\omega g_3^\sigma$, from the $\mathcal{B}$'s point of view, the decryption key is distributed uniformly at random because no G_{p_2} part is in T.
If $T = g_1^\omega g_2^\kappa g_3^\sigma$, the decryption key is semi-functional and the corresponding parameters are: $\vec{\theta} = \langle \kappa dw_1, \dots, \kappa dw_n\rangle$, $\theta_1 = -\kappa(aID_{k+1} + b + \sum_{i=1}^n x_i w_i)$, $\theta_2 = \kappa d$.
Because $w_1, \dots, w_n, x_1, \dots, x_n, d, a, b$ are merely calculated by modulo p_1 in mpk, from the $\mathcal{A}1$'s point of view, they are random by modulo p_2. Thus, the semi-functional decryption key is properly distributed.
$\mathcal{B}$ sets $H \leftarrow H+1$, adds the item $(H, ID_j, ID'_j, pk_{ID_j}, sk_{ID_j}$ $Cert_{ID_j}, dk_{ID_j}, 0, 0)$ to list $\mathcal{L}1$ and returns H to $\mathcal{A}1$.

(2) If it is the replace public key query with pk'_{ID_j}, $\mathcal{B}$ picks randomly $(z_1, z_2, \dots, z_n, r) \in Z_N^{n+1}$ and $\vec{\rho} \in Z_N^{n+2}$.
  (a) If $j \leq k$, $\mathcal{B}$ randomly selects a vector $\vec{\gamma} \in Z_N^{n+2}$ and calculates the certificate $Cert_{ID_j} = (\langle \nu_1^{z_1}, \dots, \nu_1^{z_n}\rangle, g_1^\alpha(u_1^{ID_j} h_1)^{-r} \prod_{i=1}^{n} g_1^{-x_i z_i}, \nu_1^r) * (g_2^\mu g_3^\rho)^{\vec{\gamma}} * g_3^{\vec{\rho}}$.
  (b) If $j > k+1$, $\mathcal{B}$calculates the certificate $Cert_{ID_j} = (\langle \nu_1^{z_1}, \dots, \nu_1^{z_n}\rangle, g_1^\alpha(u_1^{ID_j} h_1)^{-t} \prod_{i=1}^{n} g_1^{-x_i z_i}, \nu_1^r) * g_3^{\vec{\rho}}$.
  (c) If$j = k+1$, by using the challenge item T the challenger $\mathcal{B}$ calculates the certificate $Cert_{ID_{k+1}} = (\langle T^{dz_1}, \dots, T^{dz_n}\rangle, T^{-(aID_{k+1}+b)} \prod_{i=1}^{n} T^{-x_i z_i} * g_1^\alpha, T^d) * g_3^{\vec{\rho}}$.

If $T = g_1^\omega g_3^\sigma$, from the $\mathcal{B}$'s point of view, the certificate is properly distributed because no G_{p_2} part is in T.

If $T = g_1^\omega g_2^\kappa g_3^\sigma$, the certificate is semi-functional and we set $\vec{\eta} = \langle \kappa dz_1, \dots, \kappa dz_n\rangle$, $\eta_1 = -\kappa(aID_{k+1} + b + \sum_{i=1}^n x_i z_i)$ and $\eta_2 = \kappa d$.

Because $z_1, \dots, z_n, x_1, \dots, x_n, d, a, b$ are merely calculated by modulo p_1 in mpk, from the $\mathcal{A}1$'s point of view, they are random by modulop_2. Thus, the SF certificate is properly distributed.

$\mathcal{B}$ sets $H \leftarrow H+1$, adds the item $(H, ID_j, ID'_j, pk'_{ID_j}, \perp, Cert_{ID_j}, \perp, 0, 0)$ to list $\mathcal{L}1$, and returns H to $\mathcal{A}1$.

Challenge: $\mathcal{A}1$ gives an identity $ID*$ and two messages M_0 and M_1. $\mathcal{B}$ selects a random bit $\xi \in \{0, 1\}$. $\mathcal{B}$ scans the list $\mathcal{L}1$ to find $pk_{ID_j} = (Y, \pi)$ about $ID*$ for the largest handle where $Y = e(g_1, \nu_1)^{\alpha\beta^*}$. If the $\mathcal{A}1$ does not replace the public key $\mathcal{B}$ knows $\beta*$. Otherwise, $\mathcal{B}$can extract $\beta*$ from π by NIZK proof. By using $g_1^z g_2^\nu$ $\mathcal{B}$ outputs the SF ciphertext $C^* = (C_0^*, \vec{C}^*, C_1^*, C_2^*) = (M_\xi \cdot e(\nu_1^{\beta^*}, g_1^z g_2^\nu)^\alpha, \langle (g_1^z g_2^\nu)^{x_1}, \dots, (g_1^z g_2^\nu)^{x_n}\rangle, (g_1^z g_2^\nu)^d, (g_1^z g_2^\nu)^{aID^*+b})$.

From the SF ciphertext we get SF parameters $\vec{\delta} = \langle \nu x_1, \dots, \nu x_n\rangle$, $\delta_1 = \nu d$, $\delta_2 = \nu(aID^* + b)$. For the same reason as before, from the $\mathcal{A}1$'s point of view, $x_1, \dots, x_n, d, a, b$ are random under modulo p_2 and the given ciphertext is distributed uniformly at random if $ID*$ is not the identity for the $(k+1)th$ handle.

(I) If the $(k+1)th$ inquiry is create inquiry, we can get the conclusion: If T contains G_{p_2} part, the decryption key toward the $(k+1)th$ handle is semi-functional for the $C*$. The NSF property is embodied as follows. $\vec{\theta} \cdot \vec{\delta} = \kappa\nu d \sum_{i=1}^n x_i w_i \mod p_2$, $\theta_1 \cdot \delta_1 = -\kappa\nu d(aID_{k+1} + b + \sum_{i=1}^n x_i w_i) \mod p_2$ and $\theta_2 \cdot \delta_2 = -\kappa\nu d(aID^* + b) \mod p_2$.

If $T = g_1^w g_2^\kappa g_3^\sigma$, $\mathcal{B}$ imitates the $\mathcal{T}1\_AltGame\_k$ correctly when $ID^* \neq ID_{k+1} \mod p_2$. Or else, $\mathcal{B}$ imitates the $\mathcal{T}1\_Game\_k$.

(II) If the $(k+1)th$ inquiry is $\mathcal{O} - Replacepublickey$, we can get the conclusion: Though the public key is replaced, the decryption key has the same G_{p_2} part with the corresponding certificate. The reason is that if $\mathcal{A}1$ does not affect the G_{p_1} part $\mathcal{A}1$ cannot randomize G_{p_2} part. The NSF property is embodied as follows.

$\vec{\theta} \cdot \vec{\delta} = \vec{\eta}^\beta \cdot \vec{\delta} = \kappa\nu\beta d \sum_{i=1}^n x_i w_i \mod p_2$, $\theta_1 \cdot \delta_1 = \eta_1^\beta \cdot \delta_1 = -\kappa\nu\beta d(aID_{k+1} + b + \sum_{i=1}^n x_i w_i) \mod p_2$ and $\theta_2 \cdot \delta_2 = \eta_2^\beta \cdot \delta_2 = -\kappa\nu\beta d(aID^* + b) \mod p_2$.
If $T = g_1^w g_2^\kappa g_3^\sigma$ $\mathcal{B}$ imitates the $\mathcal{T}1\_AltGame\_k$ correctly when $ID^* \neq ID_{k+1} \mod p_2$. Or else, $\mathcal{B}$ imitates the $\mathcal{T}1\_Game\_k$.

In the above two cases, NSF is taken into account as follows.

(1) $ID^* = ID_{k+1} \mod p_2$ *and* $ID^* \neq ID_{k+1} \mod N$.
(2) $ID^* = ID_{k+1} \mod N$.

For the case (1), if $\mathcal{B}$ computes $a' = \gcd(ID^* - ID_{k+1}, N)$ it can generate a nontrivial factor of N. Subgroup decision assumption is broken, which implies that Assumptions 1–3 are broken.

For the case (2), the challenge identity $ID*$ is pointed out by the $(k+1)th$ handle. In the case, $\mathcal{A}1$ cannot extract the decryption key of $ID*$. We further divide it into two subcases:

(a) $\mathcal{A}1$ may obtain the certificate and some leakage of the decryption key for identity $ID*$.
(b) $\mathcal{A}1$ is able to replace the corresponding public key and get some leakage of the certificate for identity $ID*$.

Lemma 5 is to show that normal SF and NSF are indistinguishable.

Lemma 5. *For the case (2) of Lemma 4, when the decryption key/certificate changes from normal SF to NSF for the $(k+1)th$ handle of* *identity ID∗ in the case of λ_{dk} leakage of the decryption key, the advantage that $\mathcal{A}1$ wins is a negligible value, where $\lambda_{dk} = (n-2c-1)\lambda$ and c is a fixed positive constant.*

Proof. If there is a PPT adversary $\mathcal{A}1$ who distinguishes the SF and NSF decryption key or certificate with non-negligible advantage, we can construct an algorithm $\mathcal{B}$ to break Corollary 1. That is to say, the advantage that $\mathcal{B}$ succeeds in distinguishing the distribution $(\vec{\delta}, f(\vec{\tau}'))$ and $(\vec{\delta}, f(\vec{\tau}))$ is non-negligible. Thus, there is a contradiction.

$\mathcal{B}$ runs **Setup**. $\mathcal{B}$ keeps the *msk* and gives the *mpk* to $\mathcal{A}1$. Because $\mathcal{B}$ has the *msk* and knows $g_2 \in G_{p_2}$, it can generate normal and semi-functional keys. Thus, it can answer all the inquiries of $\mathcal{A}1$.

Toward the $(k+1)th$ replace public key or create inquiry about identity *ID*, $\mathcal{B}$ replies with the handle $H*$ and does not create the private key. If $\mathcal{A}1$ asks for the leakage of decryption key for $(H*, f)$, $\mathcal{B}$ encodes the leakage that $\mathcal{A}1$ inquiries in Phase 1 for the *ID* as a PPT function $f: Z_{p_2}^n \to 2^{\lambda_{dk}}$. This can be done if all the values of other keys and other variables for the challenge key are regarded as fixed. Then $\mathcal{B}$ gets $(\vec{\delta}, f(\vec{\Gamma}))$, where $\vec{\Gamma}$ is $\vec{\tau}$ or $\vec{\tau}'$ according to Corollary 1 and has $n+1$ components. $\mathcal{B}$ returns $f(\vec{\Gamma})$ to $\mathcal{A}1$ as the leakage about the $(k+1)th$ handle, which defines the challenge keys in the following.

(1) If the $(k+1)th$ inquiry is create query, $\mathcal{B}$calculates normal certificate $Cert_{ID}$ by using *msk*. $\mathcal{B}$ selects randomly $r_1, r_2 \in Z_{p_2}$ and sets implicitly G_{p_2} part in decryption key to be $g_2^{\vec{\Gamma}'}$, where $\vec{\Gamma}'$ is defined as $\langle \vec{\Gamma}, 0\rangle + \langle \underbrace{0, \ldots, 0}_{n}, r_1, r_2\rangle$. $\mathcal{B}$ sets non-G_{p_2}part in decryption key to satisfy the proper distribution. It is able to solve the subcase (a) of Lemma 4.

(2) If the $(k+1)th$ inquiry is $\mathcal{O}-Replacepublickey$, $\mathcal{B}$ selects randomly $r_1, r_2 \in Z_{p_2}$and sets G_{p_2} part in the certificate to be $g_2^{\vec{\Gamma}'}$, where $\vec{\Gamma}'$is defined as $\langle \vec{\Gamma}, 0\rangle + \langle \underbrace{0, \ldots, 0}_{n}, r_1, r_2\rangle$. $\mathcal{B}$ sets non-G_{p_2} part in the certificate to satisfy the proper distribution. It is able to solve the subcase (b) of Lemma 4.

In certain circumstances, $\mathcal{A}1$ gives the challenge message. $\mathcal{B}$ selects $t_2 \in Z_{p_2}$which satisfies $\delta_n r_1 + t_2 r_2 \equiv 0 \mod p_2$. By using $\langle \vec{\delta}, 0\rangle + \langle 0, \ldots, 0, 0, t_2\rangle$ as G_{p_2} part where $\vec{\delta}$ has $n+1$ components, $\mathcal{B}$ generates the challenge ciphertext. If $\vec{\delta}$ and $\vec{\Gamma}$ are orthogonal, the $(k+1)th$ handle is about the NSF decryption key/certificate.

It is obvious that $\mathcal{B}$ is able to reply easily the inquiries in Phase 2. By using the output of $\mathcal{A}1$, $\mathcal{B}$ succeeds in distinguishing the distribution $(\vec{\delta}, f(\vec{\tau}'))$ and $(\vec{\delta}, f(\vec{\tau}))$ with non-negligible advantage.

If Assumption 2 holds the $\mathcal{T}1\_Game\_k$ and $\mathcal{T}1\_AltGame\_k$ are indistinguishable from the adversary's point of view. Namely, $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_AltGame\_k}(\lambda_{dk}, \lambda_{msk})| \le \varepsilon$.

Lemma 6. *If $\lambda_{dk} = (n-2c-1)\lambda$ and Assumption 2 holds, $\mathcal{T}1\_AltGame\_k$ and $\mathcal{T}1\_Game\_k+1$ are indistinguishable from the adversary's point of view. That is $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_AltGame\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k+1}(\lambda_{dk}, \lambda_{msk})| \le \varepsilon$.*

Proof. If there is a PPT adversary $\mathcal{A}1$ who distinguishes the $\mathcal{T}1\_AltGame\_k$ and $\mathcal{T}1\_Game\_k+1$ with non-negligible advantage, we can construct a simulator $\mathcal{B}$ who breaks Assumption 2. Firstly, $\mathcal{B}$ is given an instance $(N, G, G_T, e, g_1, g_3, g_1^z g_2^v, g_2^u g_3^\rho)$. $\mathcal{B}$ is given a challenge item T which is $g_1^\omega g_3^\sigma$ or $g_1^\omega g_2^\kappa g_3^\sigma$. $\mathcal{B}$ interacts with $\mathcal{A}1$ as follows.

Setup phase: $\mathcal{B}$ firstly creates composite order bilinear groups $(N = p_1p_2p_3, G, G_T, e)$. $\mathcal{B}$ selects $a, b, d \in Z_N$ uniformly at random and sets $u_1 = g_1^a$, $h_1 = g_1^b$ and $v_1 = g_1^d$. Given $g_1 \in G_{p_1}$ and $g_3 \in G_{p_3}$, $\mathcal{B}$ runs the algorithm *Gen* of $\prod$ to generate the common reference string *crs* and selects random$(\alpha, x_1, x_2, \ldots, x_n, r, y_1, y_2, \ldots, y_n) \in Z_N^{2n+2}$ and a vector $\vec{\rho} = \langle \rho_1, \rho_2, \ldots, \rho_{n+3}\rangle \in Z_N^{n+3}$ where $n \ge 2$ is an integer.

It outputs the master public key $mpk = (N, G, G_T, e, e(g_1, v_1)^\alpha, g_1, g_1^{x_1}, \ldots, g_1^{x_n}, u_1, h_1, v_1, g_3, crs)$ and the master secret key $msk = (\langle v_1^{y_1}, \ldots, v_1^{y_n}\rangle, g_1^\alpha h_1^{-r} \prod_{i=1}^{n} g_1^{-x_i y_i}, v_1^r, u_1^r) * g_3^{\vec{\rho}} \triangleq (\vec{K}, K_1, K_2, K_3)$.

Oracle query: $\mathcal{B}$ may reply every query because it knows α, x_i, y_i, r. Specifically, $\mathcal{B}$ answers the jth query for identity ID_j as follows.

(1) If it is the create query, $\mathcal{B}$ generates normal certificate $Cert_{ID_j} = (\vec{D}, D_1, D_2)$ by using *msk*. The $Cert_{ID_j}$ can leak information. $\mathcal{B}$ runs **SetPrivateKey** to get β and runs **SetPublicKey** to get pk_{ID_j}.
  (a) If $j \le k$, $\mathcal{B}$ selects randomly vectors $\vec{\rho}_1, \vec{\gamma}_1, \vec{\rho}_2, \vec{\gamma}_2 \in Z_N^{n+2}$ and calculates the decryption key: $\widetilde{Cert_{ID}} = (\vec{D}, D_1, D_2) * (g_2^\mu g_3^\rho)^{\vec{\gamma}_1} * g_3^{\vec{\rho}_1}$, $\widetilde{dk_{ID}} = (\vec{S}, S_1, S_2) * (g_2^\mu g_3^\rho)^{\vec{\gamma}_2} * g_3^{\vec{\rho}_2}$. $\mathcal{B}$ adds $(H, ID_j, ID_j', pk_{ID_j}, sk_{ID_j}, \widetilde{Cert_{ID_j}}, \widetilde{dk_{ID_j}}, 0, 0)$ to the list $\mathcal{L}1$.
  (b) If $j > k+1$, $\mathcal{B}$ calculates normally the decryption key dk_{ID_j} and adds $(H, ID_j, ID_j', pk_{ID_j}, sk_{ID_j}, Cert_{ID_j}, dk_{ID_j}, 0, 0)$ to the list $\mathcal{L}1$.
  (c) If $j = k+1$, $\mathcal{B}$ selects randomly vectors $\vec{\rho}_1, \vec{\rho}_2, \vec{\gamma}_1 \in Z_N^{n+2}$ and $(z_1, z_2, \ldots, z_n) \in Z_N^n$. By using the challenge item $\mathcal{B}$ calculates the certificate $Cert_{ID_{k+1}} = (\langle T^{d_1 z_1}, \ldots, T^{d_1 z_n}\rangle, T^{-(aID_{k+1}+b)} \prod_{i=1}^{n} T^{-x_i z_i} * g^\alpha, T^d) * g_3^{\vec{\rho}_1}$.

$\mathcal{B}$ invokes **SetDecryptKey** to get decryption key dk_{ID_j} and calculates SF decryption key $\widetilde{dk_{ID_j}} = (\vec{S}, S_1, S_2) * (g_2^\mu g_3^\rho)^{\vec{\gamma}_2} * g_3^{\vec{\rho}_2}$. $\mathcal{B}$ adds $(H, ID_j, ID_j', pk_{ID_j}, sk_{ID_j}, Cert_{ID_j}, \widetilde{dk_{ID_j}}, 0, 0)$ to the list $\mathcal{L}1$.

If $T = g_1^\omega g_3^\sigma$, from the $\mathcal{B}$'s point of view, the decryption key is uniformly distributed at random because no G_{p_2} part is in T.

If $T = g_1^\omega g_2^\kappa g_3^\sigma$, the decryption key is semi-functional and the corresponding parameters are $\vec{\eta} = \langle \kappa d_1 z_1, \ldots, \kappa d_1 z_n\rangle$, $\eta_1 = -\kappa(aID_{k+1} + b + \sum_{i=1}^{n} x_i z_i)$ and $\eta_2 = \kappa d$.

Because $z_1, \ldots, z_n, x_1, \ldots, x_n, d, a, b$ are merely calculated by modulo p_1 in *mpk*, from the $\mathcal{A}1$'s point of view, they are random by modulo p_2. Thus, the semi-functional decryption key is properly distributed.

$\mathcal{B}$ sets $H \leftarrow H+1$ and returns $H+1$ to $\mathcal{A}1$.

(2) If the jth inquiry is replace public key inquiry, the simulation is run in the same way as that of Lemma 4.

Challenge: It is run in the same way as the Challenge of Lemma 4. The NSF property is analyzed in a similar way as that of Lemma 4.

To summarize, if Assumption 2 holds and the leakage amount is at most $(n-2c-1)\lambda$, $\mathcal{T}1\_Game\_k$ and $\mathcal{T}1\_AltGame\_k$ are indistinguishable from the adversary's point of view. Likewise, $\mathcal{T}1\_AltGame\_k$ and $\mathcal{T}1\_Game\_k+1$ are indistinguishable from the adversary's point of view. Thus, $\mathcal{T}1\_Game\_k$ and $\mathcal{T}1\_Game\_k+1$ are indistinguishable from the adversary's point of view.

By Lemmas 4–6, we have

$$|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k+1}(\lambda_{dk}, \lambda_{msk})|$$

$$= \Big|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_AltGame\_k}(\lambda_{dk}, \lambda_{msk}) + Adv_{\mathcal{A}1}^{\mathcal{T}1\_AltGame\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k+1}(\lambda_{dk}, \lambda_{msk})\Big|$$

$$\le \begin{array}{c} |Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_AltGame\_k}(\lambda_{dk}, \lambda_{msk})| \\ +|Adv_{\mathcal{A}1}^{\mathcal{T}1\_AltGame\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k+1}(\lambda_{dk}, \lambda_{msk})| \end{array}$$

$= \varepsilon + \varepsilon = 2\varepsilon$. This finishes Lemma 3. Thus, we have

$$|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_k+1}(\lambda_{dk}, \lambda_{msk})| \le \varepsilon.$$

Lemma 7. *If* $\lambda_{msk} = (n - 2c - 1)\lambda$ *and* Assumption 2 *holds,* $\mathcal{T}1\_Game\_Q$ *and* $\mathcal{T}1\_Game\_Msk$ *are indistinguishable from the adversary's point of view. That is* $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Q}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Msk}(\lambda_{dk}, \lambda_{msk})| \le \varepsilon$.

Proof. If there is a PPT adversary $\mathcal{A}1$ who may distinguish the $\mathcal{T}1\_Game\_Q$ and $\mathcal{T}1\_Game\_Msk$ with non-negligible advantage, we can construct a simulator $\mathcal{B}$ who breaks Assumption 2. Firstly, $\mathcal{B}$ is given an instance $(N, G, G_T, e, g_1, g_3, g_1^z g_2^v, g_2^u g_3^\rho)$. $\mathcal{B}$ is given a challenge item T which is $g_1^\omega g_3^\sigma$ or $g_1^\omega g_2^\kappa g_3^\sigma$. $\mathcal{B}$ interacts with $\mathcal{A}1$ as follows.

Setup phase: $\mathcal{B}$ selects $\alpha, a, b, d, x_1, \ldots, x_n \in Z_n$ and sets $g_1 = g_1, u_1 = g_1^a, h_1 = g_1^b$ and $v_1 = g^d$. $\mathcal{B}$ issues the $mpk = (N, G, G_T, e, g_1, u_1, h_1, q_1, v_1, e(g_1, v_1)^\alpha, g_1^{x_i})$. $\mathcal{B}$ selects $(y_1, \ldots, y_n) \in Z_N^n$ and $\vec{\rho} \in Z_N^{n+3}$. By using the challenge item $\mathcal{B}$ generates the master secret key $msk = (\langle T^{dy_1}, \ldots, T^{dy_n}\rangle, g^\alpha T^{-b} \prod_{i=1}^{n} T^{-x_i y_i}, T^d, T^a) * g_3^{\vec{\rho}}$.

If $T = g_1^\omega g_3^\sigma$, from the $\mathcal{B}$'s point of view, the msk is properly distributed because no G_{p_2} component is in T.

If $T = g_1^\omega g_2^\kappa g_3^\sigma$, the msk is semi-functional and the corresponding parameters are $\vec{\iota} = \langle \kappa d y_1, \ldots, \kappa d y_n\rangle$, $\iota_1 = -\kappa(b + \sum_{i=1}^n x_i y_i)$, $\iota_2 = \kappa d$ and $\iota_3 = \kappa a$.

Because $y_1, \ldots, y_n, x_1, \ldots, x_n, d, a, b$ are merely calculated by modulo p_1 in mpk, from the $\mathcal{A}1$'s point of view, they are random by modulo p_2. Thus, the SF msk is distributed uniformly at random.

Oracle query: $\mathcal{B}$ uses $g_2 g_3$ to re-randomize the $G_{p_2 p_3}$. Similarly, $\mathcal{B}$ uses g_3 to re-randomize the G_{p_3}. Thus, $\mathcal{B}$ is able to create the corresponding semi-functional keys by using the semi-functional master secret key.

Challenge: The phase is the same as that of Lemma 4.

Because the msk is NSF for the challenge ciphertext, we must make further efforts to prove that NSF and SF msk are indistinguishable from the $\mathcal{A}1$'s point of view in the case of the leakage of msk.

Lemma 8. *The advantage that* $\mathcal{A}1$ *wins when the msk is adjusted from normal SF to NSF has a negligible difference in the case of* λ_{msk} *leakage of the master secret key where* $\lambda_{msk} = (n - 2c - 1)\lambda$ *and c is a fixed positive constant.*

Proof. The proof is very similar to that of Lemma 5 except for the following differences. Note that in Lemma 5 the G_{p_2} part of the certificate/decryption key is $(\vec{\Gamma}, 0) + (0, \ldots, 0, r_1, r_2)$. For the msk in Lemma 8 the G_{p_2} part is $(\vec{\Gamma}, 0, 0) + (0, \ldots, 0, r_1, r_2, 0) + (0, \ldots, 0, \theta)$ where $r_1, r_2, \in Z_{p_2}$.

When $\mathcal{A}1$ gives the challenge message for the challenge identity $ID*$ to $\mathcal{B}$, $\mathcal{B}$ chooses $t_2 \in Z_{p_2}$ such that $\delta_{n+1}(r_1 - ID^*\theta) + t_2 r_2 \equiv 0 \mod p_2$. By using the output of the adversary $\mathcal{A}1$ we succeed in distinguishing $(\vec{\delta}, f(\vec{\tau}'))$ and $(\vec{\delta}, f(\vec{\tau}))$ with non-negligible advantage. Thus, there is a contradiction.

By Lemmas 7 and 8, we have $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Q}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Msk}(\lambda_{dk}, \lambda_{msk})| \le \varepsilon$.

Lemma 9. *Under* Assumption 3 *the advantage that PPT adversary succeeds in distinguishing* $\mathcal{T}1\_Game\_Msk$ *and* $\mathcal{T}1\_Game\_Final$ *is negligible. That is* $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Msk}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk}, \lambda_{msk})| \le \varepsilon$.

Proof. If there is a PPT adversary $\mathcal{A}1$ who may succeed in distinguishing $\mathcal{T}1\_Game\_Msk$ and $\mathcal{T}1\_Game\_Final$ with non-negligible advantage, we are able to construct a simulator $\mathcal{B}$ to break Assumption 3. $\mathcal{B}$ is given an instance $(N, G, G_T, e, g_1, g_2, g_3, g_1^\alpha g_2^v, g_1^s g_2^u)$. $\mathcal{B}$ is given a challenge item T which is $g_1^{\alpha s}$ or a random value in G. $\mathcal{B}$ interacts with $\mathcal{A}1$ as follows.

Setup: $\mathcal{B}$ selects randomly $\alpha, x_1, \ldots, x_n, a, b, d \in Z_N$ and sets the master public key $mpk = (e(g_1^\alpha g_2^v, v_1), g_1^{x_i}, u_1 = g_1^a, h_1 = g_1^b, v_1 = g_1^d)$. $\mathcal{B}$ selects randomly$y_1, \ldots, y_n, r \in Z_N$ and two vectors $\vec{\rho}, \vec{\gamma} \in Z_N^{n+3}$ and calculates the SF $msk = (\langle v_1^{y_1}, \ldots, v_1^{y_n}\rangle, h_1^{-r} \prod_{i=1}^{n} g_1^{-x_i y_i}, v_1^r, u_1^r) * g_2^{\vec{\gamma}} * g_3^{\vec{\rho}}$.

Oracle query: By using the SF master secret key, $\mathcal{B}$ simulates all oracles.

Challenge: $\mathcal{A}1$ gives an identity $ID*$ and two messages M_0 and M_1 to $\mathcal{B}$. $\mathcal{B}$ selects a random bit $\xi \in \{0, 1\}$. $\mathcal{B}$ scans the list $\mathcal{L}1$to find $pk_{ID^*} = (Y, \pi)$ about $ID*$ for the largest handle where $Y = e(g_1, v_1)^{\alpha\beta^*}$. If the $\mathcal{A}1$ does not replace the public key $\mathcal{B}$ knows $\beta*$. Otherwise, $\mathcal{B}$ can extract $\beta*$ from π by NIZK proof. By using $g_1^s g_2^u$ and T given from the assumption $\mathcal{B}$ outputs the SF ciphertext $C^* = (C_0^*, \vec{C}^*, C_1^*, C_2^*) = (M_\xi^* \cdot e(T, v_1^{\beta^*}), (g_1^s g_2^\mu)^{x_i}, (g_1^s g_2^\mu)^d, (g_1^s g_2^\mu)^{aID^*+b})$.

From the $\mathcal{A}1$'s point of view, $x_1, \ldots, x_n, d, a, b$ are random under modulo p_2. So, the given SF ciphertext is distributed uniformly at random.

If $T = g_1^{\alpha s}$, the ciphertext is distributed uniformly at random. In this case, $\mathcal{B}$ correctly simulates the $\mathcal{T}1\_Game\_Msk$. If T is a random element of G, the item C_0^* is random. Thus the ciphertext is SF for a random message. In this case, $\mathcal{B}$ correctly simulates the $\mathcal{T}1\_Game\_Final$. $\mathcal{B}$ can break Assumption 3 by using the output of $\mathcal{A}1$. There is a contraction. Thus, we have $|Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Msk}(\lambda_{dk}, \lambda_{msk}) - Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk}, \lambda_{msk})| \le \varepsilon$.

All in all, under the above nine lemmas we have that $\mathcal{T}1\_Game\_R$ and $\mathcal{T}1\_Game\_Final$ are indistinguishable from the adversary's point of view. What is more, $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_Final}(\lambda_{dk}, \lambda_{msk}) \le \varepsilon$ is proved in Theorem 6.8 of the full version of Lewko et al. (2011). To sum up, $Adv_{\mathcal{A}1}^{\mathcal{T}1\_Game\_R}(\lambda_{dk}, \lambda_{msk}) \le \varepsilon$. In addition, Lemma 1 proves the leakage bound. Thus, we finish the proof of Theorem 1.

5.3.2. Security proof against type II adversary

Theorem 2. *Under* Assumptions 1–3, *the LR-CBE scheme is secure in the case of* λ_{dk} *leakage of the decryption key against* $\mathcal{A}2$, *where the amount of the leakage is at most* $\lambda_{dk} = (n - 2c - 1)\lambda$.

Proof. The proof of Theorem 2 is similar to that of Theorem 1. Due to the space limitation, we give the concrete proofs in the full paper (Yu et al., 2015).

6. Leakage bound

Our scheme is resilient to the λ_{msk} leakage of the master secret key and the λ_{dk} leakage of the decryption key. The λ_{msk} and λ_{dk} have the same maximum value $(n - 2c - 1)\lambda$, where $n \ge 2$ is an integer and c is a fixed positive constant. The leakage is subject to the size of the subgroup G_{p_2}. The value of n can be varied.

In our system $N = p_1 p_2 p_3$ and p_1, p_2, p_3 are λ-bit primes. The size of the master secret key is $(n+3)(\lambda + \lambda + \lambda) = 3(n+3)\lambda$. Similarly, the size of the decryption key is $3(n+2)\lambda$. The leakage rate of master secret key is $\frac{(n-2c-1)\lambda}{3(n+3)\lambda} = \frac{(n-2c-1)}{3(n+3)}$. The leakage rate of decryption key is $\frac{(n-2c-1)\lambda}{3(n+2)\lambda} = \frac{(n-2c-1)}{3(n+2)}$.

The value of n can be varied and thus we obtain variable key size and variable leakage amount. The bigger value of n will allow us to get the higher leakage rate. Thus we can achieve the stronger security. The smaller value of n will lead to the smaller master public key. The leakage rate is close to 1/3 if n is large enough.

Theoretically speaking, we can get a very high leakage rate when n is large enough. The leakage rate of our scheme can come up to 1/3 which far exceeds that of the scheme (Naor and Segev, 2012). In view of engineering practice, we hope that n is a small value. We can see that when $n = 4$ the leakage rate of our scheme can come up to 1/6 which already reaches that of the scheme (Naor and Segev, 2012). What is more, we need only 6 pairings in our decryption which is acceptable in decryption calculation. In some settings, if the leakage is not serious we can even set $n = 2$. Even so, the leakage rate of our

Table 3
Performance comparison of our scheme and schemes in Gentry (2003) and Li et al. (2012).

Schemes	Encryption calculation	Decryption calculation	Leakage resilience
BasicCBE in Gentry (2003)	$3H+2P$	$1H+1P$	No
FullCBE in Gentry (2003)	$4H+2P$	$3H+1P$	No
Scheme in Li et al. (2012)	$1H+1P$	$1H+1P$	No
Our scheme	$1H+1P$	$1H+(n+2)P$	Yes

Table 4
The changes of leakage rate and decryption calculation about the parameter n.

n	2	3	4	5	…	n	…	$+\infty$
Pairings of decryption	4	5	6	7	…	$n+2$	…	$+\infty$
Leakage rate	$\frac{1}{12}$	$\frac{2}{15}$	$\frac{1}{6}$	$\frac{4}{21}$	…	$\frac{n-1}{3(n+2)}$	…	$\frac{1}{3}$

scheme is 1/12 which is slightly good. Thus, our scheme is also practically usable if we select the small n.

7. Comparisons

We compare our scheme with the schemes in Gentry (2003) and Li et al. (2012). There are two CBE schemes in Gentry (2003), BasicCBE and FullCBE, neither of which has leakage resilience. The major contribution of Li et al. (2012) is a key encapsulation mechanism which can be used to construct CBE. The key encapsulation mechanism has no leakage resilience either. Our scheme is a practical and secure scheme with leakage resilience. Denote the hash operation by H and the pairing computation by P. The calculations of encryption and decryption of the above schemes are given in Table 3.

As shown in Table 3, for the encryption calculation, our scheme is as good as that of the work in Li et al. (2012).

What is more, the value of n has an important impact on the decryption calculation of our scheme. The value of n also determines the key leakage rate. From Table 3, we can see that a bigger n will lead to more decryption operations in our scheme. But as shown in Section 6, a bigger n will help us to achieve a higher leakage rate (and stronger security). In practice, we should make a trade off between security and computational overhead in encryption and decryption.

When n varies, Table 4 gives the corresponding changes of the decryption calculation and leakage rate.

We can see from Table 3 that the decryption needs $n+2$ pairings. According to Table 4, we will know the following fact easily. When n is very large, both the decryption overhead and the leakage rate are very large. On the other hand, when the n is smaller, the key leakage rate is smaller and accordingly the decryption overhead is smaller. Even $n=2$ is a very small valve, the key leakage rate may come to $\frac{1}{12}$ which can afford the better leakage resilience in practice. At the same time, when $n=2$ the decryption only needs 4 pairings which is acceptable in practical application. To say the least, when $n=4$ the leakage rate may amount to $\frac{1}{6}$ which affords a very good leakage resilience (refer to Naor and Segev, 2012), but even so the decryption needs no more than 6 pairings which is also acceptable for engineering practice. Thus, we may select smaller value of n for the practicality, such as $n=2$, $n=3$ or $n=4$. At the same time, the leakage resilience of our scheme is very good.

8. Conclusion

Formal definitions and security models for LR-CBE are given in this paper. We present a leakage-resilient certificate-based encryption scheme in which leakage about the decryption key and the master secret key is considered. The security of the scheme is reduced to the composite order bilinear groups assumption. To the best of our knowledge, this is the first LR-CBE resilient to master secret key leakage. Our scheme has good leakage resilience. The leakage rate is close to 1/3 if we adjust n properly. Performance analysis shows our scheme has low computation overhead in encryption phase. To improve efficiency for decryption operation, we can select $n=2$ and decryption operation only needs 4 pairings which is acceptable in practical applications. As a direction of future work, we will construct secure LR-CBE under standard complexity assumptions such as prime order bilinear groups assumption, which can make the scheme more efficient.

Acknowledgments

We would like to thank anonymous referees for their helpful comments and suggestions to improve our paper. This research is supported by the National Natural Science Foundation of China (61272542, 61472083, 61402110, 61170298), the Fundamental Research Funds for the Central Universities (2013B07014), the Funding of Jiangsu Innovation Program for Graduate Education (KYZZ_0139) and the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (14KJD520006), Fok Ying Tung Education Foundation, Ministry of Education, China (141065), Ph.D. Programs Foundation of Ministry of Education of China (20123503120001), Program for New Century Excellent Talents in Fujian University (JA14067), Distinguished Young Scholars Fund of Department of Education, Fujian Province, China (JA13062).

References

Akavia, A., Goldwasser, S., Vaikuntanathan, V., 2009. Simultaneous hardcore bits and cryptography against memory attacks. In: Reingold, O. (Ed.), TCC. Vol. 5444: Lecture Notes in Computer Science. Springer, pp. 474–495.
Alwen, J., Dodis, Y., Naor, M., et al., 2010. Public-key encryption in the bounded-retrieval model. In: Gilbert, H. (Ed.), EUROCRYPT. Vol. 6110: Lecture Notes in Computer Science. Springer, pp. 113–134.
Alwen, J., Dodis, Y., Wichs, D., 2009. Leakage-resilient public-key cryptography in the bounded-retrieval model. In: Halevi, S. (Ed.), CRYPTO. Vol. 5677: Lecture Notes in Computer Science. Springer, pp. 36–54.
Boneh, D., Goh, E., Nissim, K., 2005. Evaluating 2-DNF formulas on ciphertexts. In: Kilian, J. (Ed.), TCC. Vol. 3378: Lecture Notes in Computer Science. Springer, pp. 325–341.
Brakerski, Z., Kalai, Y.T., Katz, J., Vaikuntanathan, V., 2010. Overcoming the hole in the bucket: public-key cryptography resilient to continual memory leakage. FOCS. IEEE, pp. 501–510.
Brumley, D., Boneh, D., 2005. Remote timing attacks are practical. Comput. Networks 48 (5), 701–716.
Chen, C., Wang, T., Tian, J., 2013. Improving timing attack on RSA-CRT via error detection and correction strategy. Inf. Sci. 232, 464–474.
Chen, Y., Luo, S., Chen, Z., 2011. A new leakage-resilient IBE scheme in the relative leakage model. In: Li, Y. (Ed.), Data and Applications Security and Privacy XXV. Vol. 6818: Lecture Notes in Computer Science. Springer, pp. 263–270.
Chow, S.M., Dodis, Y., Rouselakis, Y., Waters, B, 2010. Practical leakage-resilient identity-based encryption from simple assumptions. CCS. ACM, pp. 152–161.
Dodis, Y., Pietrzak, K., 2010. Leakage-resilient pseudorandom functions and side-channel attacks on Feistel networks. In: Rabin, T. (Ed.), CRYPTO. Vol. 6223: Lecture Notes in Computer Science. Springer, pp. 21–40.
Dziembowski, S., Pietrzak, K., 2008. Leakage-resilient cryptography. FOCS. IEEE, pp. 293–302.
Faust, S., Kiltz, E., Pietrzak, K., et al., 2010. Leakage-resilient signatures. In: Micciancio, D. (Ed.), TCC. Vol. 5978: Lecture Notes in Computer Science. Springer, pp. 343–360.
Gandolfi, K., Mourtel, C., Olivier, F., 2001. Electromagnetic analysis: concrete results. In: Koç, C.K., Naccache, D., Paar, C. (Eds.), CHES. Vol. 2162: Lecture Notes in Computer Science Volume. Springer, pp. 251–261.
Gentry, C., 2003. Certificate-based encryption and the certificate revocation problem. In: Biham, E. (Ed.), EUROCRYPT. Vol. 2656: Lecture Notes in Computer Science. Springer, pp. 272–293.
Groth, J., 2010. Short non-interactive zero-knowledge proofs. In: Abe, M. (Ed.), ASIACRYPT. Vol. 6477: Lecture Notes in Computer Science Volume. Springer, pp. 341–358.
Halderman, J.A., Schoen, S.D., Heninger, N., et al., 2009. Lest we remember: cold-boot attacks on encryption keys. Commun. ACM 52 (5), 91–98.
Katz, J., Vaikuntanathan, V., 2009. Signature schemes with bounded leakage resilience. In: Matsui, M. (Ed.), ASIACRYPT. Vol. 5912: Lecture Notes in Computer Science. Springer, pp. 703–720.
Lewko, A., Waters, B., 2010. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In: Micciancio, D. (Ed.), TCC. Vol. 5978: Lecture Notes in Computer Science. Springer, pp. 455–479.

Lewko, B., Rouselakis, Y., Waters, B., 2011. Achieving leakage resilience through dual system encryption. In: Ishai, Y. (Ed.), TCC. Vol. 6597: Lecture Notes in Computer Science. Springer, pp. 70–88.
Li, J., Huang, X., Hong, M., Zhang, Y., 2012a. Certificate-based signcryption with enhanced security features. Comput. Math. Appl. 64 (6), 1587–1601.
Li, J., Huang, X., Mu, Y., Susilo, W, Wu, Q., 2010. Constructions of certificate-based signature secure against key replacement attacks. J. Comput. Secur. 18 (3), 421–449.
Li, J., Huang, X., Zhang, Y., Xu, L., 2012b. An efficient short certificate-based signature scheme. J. Syst. Software 85 (2), 314–322.
Li, J., Wang, Z., Zhang, Y., 2013. Provably secure certificate-based signature scheme without pairings. Inf. Sci. 233 (6), 313–320.
Li, J., Yang, H., Zhang, Y., 2012c. Certificate-based key encapsulation mechanism with tags. J. Software 23 (8), 2163–2172.
Lu, Y., Li, J., 2010. Forward-secure certificate-based encryption and its generic construction. J. Networks 5 (5), 527–534.
Lu, Y., Li, J., 2012. Constructing certificate-based encryption secure against key replacement attacks. ICIC Express Lett. Part B Appl. 3 (1), 195–200.
Luo, X., Qian, P., Zhu, Y., Wang, S., 2010. Leakage-resilient identity-based encryption scheme. In: 2010 Sixth International Conference on Networked Computing and Advanced Information Management (NCM). IEEE, pp. 324–329.
Micali, S., Reyzin, L., 2004. Physically observable cryptography. In: Naor, M. (Ed.), TCC. Vol. 2951: Lecture Notes in Computer Science. Springer, pp. 278–296.
Naor, M., Segev, G., 2012. Public-key cryptosystems resilient to key leakage. SIAM J. Comput. 41 (4), 772–814.
Pietrzak, K., 2009. A leakage-resilient mode of operation. In: Joux, A. (Ed.), EUROCRYPT. Vol. 5479: Lecture Notes in Computer Science. Springer, pp. 462–482.
Waters, B., 2009. Dual system encryption: realizing fully secure ibe and hibe under simple assumptions. In: Halevi, S. (Ed.), CRYPTO. Vol. 5677: Lecture Notes in Computer Science. Springer, pp. 619–636.
Wu, W., Mu, Y., Susilo, W., Huang, X., Xu, L., 2012. A provably secure construction of certificate-based encryption from certificateless encryption. Comput. J. 55 (10), 1157–1168.
Xiong, H., Yuen, T.H., Zhang, C., Yiu, S.M., He, Y.J., 2013. Leakage-resilient certificateless public key encryption. AsiaPKC 2013. ACM, pp. 13–22.
Yu Q., Li J., Zhang Y., Wu W., Huang X., Xiang Y., 2015. Certificate-based encryption resilient to key leakage. Cryptology ePrint Archive, report 2015/314.

Qihong Yu received his B.S. degree in mathematics from the Xuzhou Normal University, Xuzhou, China in 2001. He received his M.S. degree in computer science from the Yangzhou University, Yangzhou, China in 2006. He is currently a lecturer and pursuing the Ph.D. degree in the College of Computer and Information, Hohai University, Nanjing, China. His research interests include cryptography, network security. He has published over 10 research papers in refereed international conferences and journals.

Jiguo Li received his B.S. degree in mathematics from Heilongjiang University, Harbin, China in 1996, M.S. degree in mathematics and Ph.D. degree in computer science from Harbin Institute of Technology, Harbin, China in 2000 and 2003, respectively. During 2006.9–2007.3, he was a visiting scholar at Centre for Computer and Information Security Research, School of Computer Science & Software Engineering, University of Wollongong, Australia. During 2013.2–2014.1, he was a visiting scholar in Institute for Cyber Security in the University of Texas at San Antonio. He is currently a Professor with the College of Computer and Information, Hohai University, Nanjing, China. His research interests include cryptography and information security, cloud computing, wireless security and trusted computing etc. He has published over 100 research papers in refereed international conferences and journals. His work has been cited more than 1200 times at Google Scholar. He has served as program committee member in over 20 international conferences and served as the reviewers in over 50 international journals and conferences.

Yichen Zhang received her B.S. degree in computer science from the Qiqihar University, Qiqihar, China in 1995. She is currently a lecturer and pursuing the Ph.D. degree in the College of Computer and Information, Hohai University, Nanjing, China. Her research interests include cryptography, network security. She has published over 30 research papers in refereed international conferences and journals.

Wei Wu received her Ph.D. degree from the School of Computer Science and Software Engineering, University of Wollongong, Australia. She is currently an Associate Professor at the School of Mathematics and Computer Science, Fujian Normal University, China. Her research interests include applied cryptography and network security. She has published over 20 research papers in refereed international conferences and journals.

Xinyi Huang received his Ph.D. degree from the School of Computer Science and Software Engineering, University of Wollongong, Australia. He is currently a Professor at the School of Mathematics and Computer Science, Fujian Normal University, China, and the Co-Director of Fujian Provincial Key Laboratory of Network Security and Cryptology. His research interests include applied cryptography and network security. He has published over 100 research papers in refereed international conferences and journals. His work has been cited more than 1700 times at Google Scholar (H-Index: 23). He is an associate editor of IEEE Transactions on Dependable and Secure Computing, in the Editorial Board of International Journal of Information Security (IJIS, Springer) and has served as the program/general chair or program committee member in over 70 international conferences.

Yang Xiang received his PhD in Computer Science from Deakin University, Australia. He is currently a full professor at School of Information Technology, Deakin University. He is the Director of the Network Security and Computing Lab (NSCLab) and the Associate Head of School (Industry Engagement). His research interests include network and system security, distributed systems, and networking. He is the Chief Investigator of several projects in network and system security, funded by the Australian Research Council (ARC). He has published more than 150 research papers in many international journals and conferences. Two of his papers were selected as the featured articles in the April 2009 and the July 2013 issues of IEEE Transactions on Parallel and Distributed Systems. He has published two books, Software Similarity and Classification (Springer) and Dynamic and Advanced Data Mining for Progressing Technological Development (IGI-Global). He has served as the Program/General Chair for many international conferences. He serves as the Associate Editor of IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, Security and Communication Networks (Wiley), and the Editor of Journal of Network and Computer Applications. He is the Coordinator, Asia for IEEE Computer Society Technical Committee on Distributed Processing (TCDP). He is a Senior Member of the IEEE.