RAIRO-Theor. Inf. Appl. 53 (2019) 67–84
https://doi.org/10.1051/ita/2019001

RAIRO - Theoretical Informatics and Applications
www.rairo-ita.org

AN EFFICIENT CERTIFICATELESS MULTI-RECEIVER THRESHOLD DECRYPTION SCHEME

Ronghai Gao[1], Jiwen Zeng[1,3] and Lunzhi Deng[2,*]

Abstract. Threshold decryption allows only quorum cooperate users to decrypt ciphertext encrypted under a public key. However, such threshold decryption scheme cannot be applied well in this situation where all users have their public and private key pairs, but do not share any private keys corresponding to the public keys, such as mobile network featured with dynamic character. The direct way to achieve threshold decryption in this case is to divide the message into several pieces and then encrypt these pieces with the public keys of different users. However, this is very inefficient. Multireceiver threshold decryption scheme that could be applied efficiently in the above situation. Recently, some certificateless (ID-based) multireceiver threshold decryption (signcryption) schemes are introduced. But the bilinear pairings are used in most of the existing schemes. In this paper, we propose an efficient certificateless threshold decryption scheme using elliptic curve cryptography (ECC) without bilinear pairing. Performance analysis shows that the proposed scheme has lower computation cost than existing some threshold decryption schemes in both encryption and decryption process. Security analysis shows that our scheme is IND-CCA secure, and no one outside of selected receivers can disclose receivers identities, against the adversaries defined in CL-PKC system under the random oracle model.

Mathematics Subject Classification. 94A60.

Received April 23, 2018. Accepted January 14, 2019.

1. Introduction

In traditional public key infrastructure (PKI) exists management, distribution, and revocation of public key certificate, these lead to high calculation and communication costs. To address this problem, in 1984, Shamir [21] proposed the cryptosystem based on identity. Its main idea is that the public keys are generated by any strings while the private keys can be generated by the key generation center (KGC). In such cryptosystem, the trust problem does not exist in the certificate-based cryptosystem because binding the public key to the corresponding user is not required. Therefore, identity-based cryptography greatly lowers the needs for certificate. Boneh and Franklin [2] introduced the first practical ID-based encryption (IBE) scheme. However, there exists the inherent key escrow problem in ID-PKC. To avoid the problem of identity-based cryptography, in

Keywords and phrases: Certificateless cryptographythreshold decryptionmulti-receiver encryption confidentiality anonymity.

[1] School of Mathematical Sciences, Xiamen University, Xiamen 361005, China.
[2] School of Mathematical Sciences, Guizhou Normal University, Guiyang 550001, China.
[3] School of Mathematical Sciences, Xinjiang Normal University, Urumqi 830017, China.

* Corresponding author: denglunzhi@163.com

[1], Al-Riyami and Paterson introduce the concept of Certificateless Public Key Cryptography (CL-PKC), and they present a concrete pairing-based Certificateless Public Key Encryption (CL-PKE) scheme. This model (CL-PKC) does not require a certificate to ensure the authenticity of the public key. In the CL-PKC cryptosystem, a KGC called a semi-trusted third party generate the partial private key for an entity with its identity by using system's master-key. The entity combines some secret value with the partial key to produce its full private key.

Threshold cryptography is a branch of research in the field of cryptography, which can avoid the problem of single point failure by sharing a secret among multiple parties. Threshold cryptography has been widely used in the construction of secret sharing, threshold encryption, threshold decryption, threshold signatures, privacy protection, etc. Threshold decryption is to disperse decryption power by distributing shares of decryption keys to multiple participants. *For example* a typical application of threshold decryption is to use multiple receivers' public keys to encrypt the message to be sent, and then send the ciphertext to these receivers. In this typical application, a cooperation containing arbitrary t out of n receivers can correctly decrypt the received ciphertext, but the message is secure if the number of the receivers is less than the threshold value t. In 1998, Shoup *et al.* [22] proposed the first provably secure threshold decryption scheme in the public key cryptosystem. Latter, many threshold decryption schemes are proposed [9, 11, 25]. Hong *et al.* [9] presented fair threshold decryption schemes which either all receivers can decrypt it, or none of them can. In their scheme, semi-trusted third parties and off-line semi-trusted third parties are used for fair exchange. Although their solution uses only a simple hashed version of ElGamal encryption, their method can also be well extended to other threshold signature schemes and threshold decryption schemes. Xu *et al.* [25] proposed a new verifiable threshold decryption scheme without trusted center, there are several advantages in this scheme such as dynamic member revocation and cheat-proof.

Combining ID-based public key cryptography with threshold decryption, the notion of identity-based threshold decryption (IBTD) scheme was proposed [12]. Latter, Long *et al.* [15] proposed an IBTD scheme, and its security is reduced to the Bilinear Diffie–Hellman problem. Therefore, two bilinear pairing operations are concealed when verifying the validity of ciphertext. Chai *et al.* [4] (2006) constructed an efficient ID-based broadcast threshold decryption scheme in mobile ad hoc network, a sending node can effectively broadcast encrypted messages to multiple dynamic groups in such a way that only the groups reaching the minimum size can decrypt the received ciphertext. In the literature [3], Chai *et al.* first proposed an ID-based threshold decryption scheme without random oracles, and proved that it is selective chosen plaintext secure under the bilinear Diffie–Hellman inversion assumption. In order to guarantee that the shared decryption is performed correctly, Ju *et al.* [10] modify Chai *et al.*'s [3] scheme to ensure that all decryption shares are consistent, and presented the first mediated IBE scheme based on the bilinear Diffie–Hellman inversion assumption without random oracles. Other threshold decryption schemes can reference literature [5, 13, 17, 19, 23, 26] in the context of identity-based cryptography.

However, less attention is paid to certificateless threshold cryptosystem in the existing literature, Long *et al.* [14] presented the first certificateless threshold decryption scheme, the scheme is secure against threshold chosen-ciphertext attack. Based on the idea of [24], in 2009, Zhang [27] introduced the first certificateless threshold decryption scheme that is IND-CCA secure against chosen ciphertext attack in the standard model.

In threshold decryption scheme, if the message is divided into several pieces and then encrypt these pieces with the public keys of different users. However, this is very inefficient way. Multireceiver threshold decryption scheme that could be applied efficiently in the above situation, but to our knowledge, only a few literatures considered such schemes [5, 18, 29]. Chai *et al.* [5] proposed the first ID-based multireceiver threshold decryption scheme. Qin *et al.* [18] proposed identity-based multireceiver threshold signcryption scheme, and the chosen-ciphertext security is proved formally. Unfortunately, Zhang *et al.* [29] pointed that Qin *et al.*'s scheme is neither semantically secure against IND-CCA nor unforgeable against EUF-CMA, and proposed an improved scheme to capture the security requirement.

1.1. Our contribution

In this paper, based on encryption idea of [6], combining the certificateless cryptography and the threshold cryptography, we propose an efficient certificateless multireceiver (t, n) threshold decryption scheme (CLMRThD) using elliptic curve cryptography (ECC), which avoid both the single point of failure in the distributed networks and the inherent key escrow problem in identity-based cryptosystem. Since our scheme does not use bilinear pairing during the encryption and decryption process, the proposed multireceiver threshold decryption scheme has higher efficiency than the existing threshold decryption schemes. In the encryption and decryption, our scheme has the lowest runtime comparing with schemes [5, 14, 15, 18, 29]. Our scheme provides confidentiality of message and anonymity of receiver under the random oracle model with the difficulties of computational Diffie–Hellman problem and against the adversaries defined in CL-PKC system.

1.2. Organization

The rest of the paper is organized as follows: mathematical preliminaries are introduced in Section 2. Formal definition of Our CLMRThD scheme is presented in Section 3. Our CLMRThD scheme is proposed in Section 4. In Section 5, we give some security analysis of our CLMRThD scheme. In Section 6, we analyze the performance of the proposed CLMThD scheme. At last, some conclusions of the paper are presented.

2. Mathematical preliminaries

In this section, we introduced the Lagrange interpolating polynomial theorem and existing some intractable problems.

2.1. Polynomial interpolation

Lagrange interpolating polynomial theorem: Let

$$f(x) = \sum_{i=1}^{k} L_i(x) = \sum_{i=0}^{k-1} a_i x^i$$

be a polynomial of degree $k-1 \geq 0$ that passes through the k points $(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)$ where for each i,

$$L_i(x) = y_i \prod_{1 \leq j \neq i \leq k} \frac{x - x_j}{x_i - x_j},$$

and

$$L_i(x_i) = y_i, L_i(x_j) = 0, x_j \in \{x_1, \ldots, x_k\} \setminus \{x_i\}, i = 1, \ldots, k.$$

2.2. Computational problems and some assumptions

Here, we mainly introduce the definitions of negligible function, decision Diffie–Hellman problem, discrete logarithm (DL) problem, and some assumptions.

Let G be a cyclic group of prime order q defined on elliptic curve, P be a generator of G.

Negligible function. We call function $\omega(k)$ is negligible if there exists l_0 such that $\omega(k) \leq \frac{1}{k^l}$ for every $l \geq l_0$.

Discrete logarithm (DL) problem. Given a random instance (P, xP), where $P \in E$, and $x \in Z_q^*$, $\mathcal{A}$ computes x from (P, xP). The probability that a polynomial time-bounded algorithm $\mathcal{A}$ can solve the DL problem is defined as $Adv_{\mathcal{A}}^{DL}(k) = Pr[\mathcal{A}(P, xP) = x : P \in E; x \in Z_q^*]$

Discrete logarithm (DL) assumption. For any PPT algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{DL}(k)$ is negligible if $Adv_{\mathcal{A}}^{DL}(k) \leq \omega$, for negligible function ω.

Decision Diffie–Hellman (DDH) problem. Given (P, aP, bP, X) for some random $a, b \in Z_q^*$ and $X \in_R \{abP, Y \in G \setminus abP\}$, decide if $X = abP$ holds. The probability that a polynomial time-bounded algorithm $\mathcal{A}$ can solve the DDH problem is defined as $Adv_{\mathcal{A}}^{DDH}(k) = \mid Pr[\mathcal{A}(P, aP, bP, X) \mid X = abP] - \frac{1}{2} \mid$

Decision Diffie–Hellman assumption. For any PPT algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{DDH}(k)$ is negligible if $Adv_{\mathcal{A}}^{DDH}(k) \leq \omega$, for negligibble function ω.

3. Formal definition of the CLMRThD scheme

The CLMRThD scheme includes four kind of participants, which they are the sender of ciphertext, the private KGC, n receivers selected and decryption combiner (DC), respectively.

Let $T = \{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n\}$ be group of n receivers selected by sender, $ID = \{ID_1, ID_2, \ldots, ID_n\}$ are their group identities, $pk = \{pk_1, pk_2, \ldots, pk_n\}$ and $sk = \{sk_1, sk_2, \ldots, sk_n\}$ are group public key and the full private key, respectively. In CLMRThD scheme, sender uses public key $\{pk_1, pk_2 \cdots, pk_n\}$ and identities $\{ID_1, ID_2, \ldots, ID_n\}$ of receivers $\{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n\}$ to encrypt message m to generate ciphertext σ, and then sends the ciphertext σ to the receivers through the public channel. Every receivers $\mathcal{R}_i$ can correctly calculate decryption share μ_i from ciphertext received by using her/his private key sk_i. DC (or one of the receivers) collects t shares to decrypt ciphertext, and then sends the plaintext to every receiver. No one other than receivers selected can disclose receiver identity in group T. Figure 1 intuitively demonstrates the process of CLMRThD scheme. The definition of the t out of n CLMRThD scheme is described as follows.

In generally, a certificateless multi-receiver (t, n) threshold decryption scheme consists of a tuple *(Setup,*

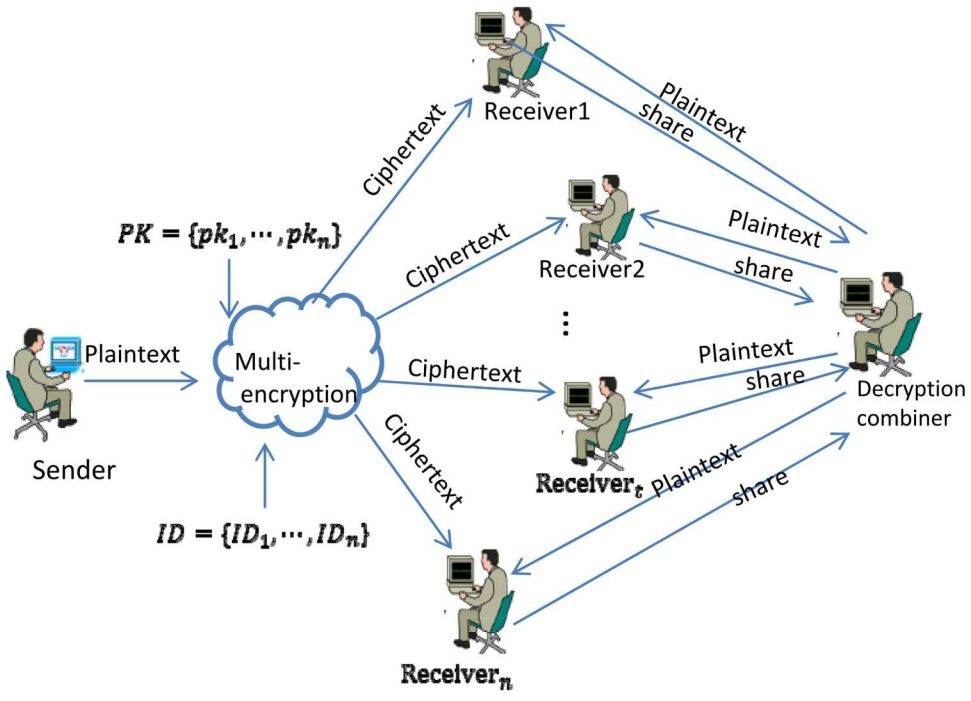

Figure 1. Process of a CLMThD scheme.

Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key, Set-Public-Key, Multi-encryption, Decryption share computation, Multi-threshold-decryption.)

- *Setup*: Run by the KGC, a security parameter k as input. It outputs the system's public parameters *params* and the master public/private key pair *(mpk,msk)*, *params and mpk* are published, and KGC keep the master private key *msk*.
- *Set-secret-value*: Run by receiver with identity ID_i himself/herself. It outputs his/her secret value r_i.
- *Set-Public-Key*: This algorithm is executed by receiver $\mathcal{R}_i$ himself/herself to generate his/her public key P_i according to his/her secret value r_i.
- *Partial-Private-Key-Extract*: Run by KGC. This algorithm takes as input the master private key *msk* and the identity ID_i of receiver $\mathcal{R}_i$, to output corresponding partial private key s_i, and delivers it to the receiver $\mathcal{R}_i$ *via* an secure channel.
- *Set-Private-Key*: This algorithm is executed by receiver $\mathcal{R}_i$ with identity ID_i. It takes (Ω, s_i, r_i) as input and returns the full private key sk_i .
- *Multi-encryption*:This is PPT algorithm. Sender executes this algorithm to generate a ciphertext for message m by identities and full public key of selected receivers .
- *decryption share computation*:Run by each receiver. This algorithm takes as input a ciphertext σ and full private key sk_i, where $i \in \{1, 2, \ldots, n\}$, and generate receiver $\mathcal{R}_i's$ decryption share μ_i.
- *Multi-threshold-decryption*: Run by $\mathcal{DC}$. This algorithm takes as input a ciphertext C, and a set of decryption shares $\{\mu_i\}_{i\in Q, |Q|\geq t}$. It outputs a plaintext M.

4. Proposed certificateless multi-receiver (t, n) threshold decryption scheme

In this section, we will present a certificateless multi-receiver (t, n) threshold decryption (CLMRThD) scheme using ECC without bilinear parings. The proposed scheme has four kinds of participants, *i.e.* a KGC, a sender S, selected n receivers $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n$, $\mathcal{DC}$. Sender encrypts message m to generate ciphertext σ for selected receivers $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n$, then sender conveys the ciphertext σ to the receivers. Every receivers $\mathcal{R}_i$ can correctly calculate decryption share μ_i from ciphertext received by using her/his private key sk_i. $\mathcal{DC}$ collects t shares to decrypt ciphertext, and then sends the plaintext to every receiver. No one other than receivers selected can disclose receiver identity. The KGC generates the system's parameter and identity-based partial private keys of all the receivers R_i for $i \in \{1, 2, \ldots, n\}$. The proposed scheme includes the following seven algorithms *(Setup, Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key, Set-Public-Key, Multi-encryption, Decryption share computation, Multi-threshold-decryption.)*

- *Setup*: k is a chosen security parameter, KGC run this algorithm to generate the system's parameters. The following steps will be implemented in this algorithm.
  (1) Choose two k-bits integers l_1, l_2, two prime integers p, q with k-bits length, and an elliptic curve E defined on F_p. Let G be additive group on elliptic curve E, and G_q be subgroup of G with prime order q.
  (2) Pick randomly a generator $P \in G_q$.
  (3) Choose $x \in_R Z_q^*$ as the master key and $P_{pub} = x \cdot P$.
  (4) Select four secure one-way resist collision hash functions $H_i : \{0,1\}^* \longrightarrow Z_q^*(i = 1, 2, 3); H_4 : \{0,1\}^* \longrightarrow \{0,1\}^{l_1+l_2}$.
  (5) Publish system's parameters $\Omega = \{p, q, l_1, l_2, E, G, G_q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ and message space $M = \{0,1\}^{l_1}$.
- *Set-secret-value*: Receiver $\mathcal{R}_i$ with identity ID_i randomly picks $r_i \in Z_q^*$ as his or her secret value and calculates $P_i = r_i \cdot P$ as the corresponding public key, and $\mathcal{R}_i$ sends (P_i, ID_i) to KGC.
- *Partial-Private-Key-Extract*: According to the identity ID_i and partial public key P_i of receiver $\mathcal{R}_i$, the KGC executes the following process:
  (1) Randomly choose $t_i \in_R Z_q^*$ and compute $T_i = t_i \cdot P$.

(2) Calculate $k_i = H_1(P_i, T_i, ID_i)$ and $s_i = t_i + k_i x \bmod (q)$.
(3) Send the tuple (T_i, s_i) to receiver R_i by authenticated secure channel.

Here, s_i is receiver $\mathcal{R}_i's$ partial private key. Partial private key s_i is valid if verify that equation $s_iP = T_i + H_1(P_i, T_i, ID_i)P_{pub}$ is true and vice versa. Since we have

$$\begin{aligned} & T_i + H_1(P_i, T_i, ID_i)P_{pub} \\ &= t_iP + k_iP_{pub} \\ &= t_iP + k_ixP \\ &= (t_i + k_ix)P \\ &= s_iP \end{aligned}$$

– *Set-Private-Key*: Receiver $\mathcal{R}_i$ secret keeps $sk_i = (r_i, s_i)$ as his or her the full private.
– *Set-Public-Key*: Receiver $\mathcal{R}_i$ keeps $pk_i = (T_i, P_i)$ as full public key.
– *Multi-encryption*: This algorithm is executed by sender $\mathcal{S}$ to generate a ciphertext of message m for selected n receivers $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n$ with identity $ID_1, ID_2, \cdots ID_n$. The following steps will be performed in this algorithm.

(1) Choose randomly $\gamma \in \{0,1\}^{l_2}$ and given message $m \in M$. Calculate $s = H_2(m, \gamma)$ and $S = sP$.
(2) Compute $U_i = s \cdot (P_i + T_i + k_iP_{pub})$ and $\mu_i = H_3(U_i, ID_i, pk_i)$,where $i = 1, 2, \ldots, n$.
(3) Randomly select $a_0, a_1, \ldots, a_{t-1} \in_R Z_q^*$ and construct a polynomial $f(x)$ with degree $t-1$ as follows:

$$\begin{aligned} (x) &= \sum_{i=0}^{t-1} a_i x^i (\mathrm{mod} q) \\ &= a_{t-1}x^{t-1} + a_{t-2}x^{t-2} + \cdots + a_1 x + a_0. \end{aligned}$$

(4) Compute $\nu_i = f(\mu_i), i = 1, 2, \ldots, n$.
(5) Compute $C = H_4(S, a_0) \oplus (m \mid\mid \gamma)$.
(6) Generate ciphertext.

$$\sigma = (S, C, \nu_1, \nu_2, \ldots, \nu_n, \Gamma),$$

where Γ is a label that contains the information about how ν_i is associated with each receiver $\mathcal{R}_i$.

– *Multi-threshold-decryption*:
Compute decryption share: To compute a decryption share μ_i of the ciphertext $\sigma = (S, C, \nu_1, \nu_2, \ldots, \nu_n, \Gamma)$ using receiver $\mathcal{R}_\rangle{}'s$ private key $sk_i = (s_i, r_i)$, receiver $\mathcal{R}_i$ calculates:
$U_i = (s_i + r_i)S$ and $\mu_i = H_3(U_i, ID_i, pk_i), i \in \{1, 2, \ldots, n\}$.
Combine A DC (or one of the receivers) collects t decryption shares $\mu_i \in Z_q^*$ to generate the plaintext, DC will perform following steps:

(1) Construct a polynomial $F(x) = \sum_{i=1}^{t} L_i(x)$ of degree $t-1$ that passes through the t points $(\mu_1, \nu_1), (\mu_2, \nu_2), \ldots, (\mu_t, \nu_t)$, where $L_i(x) = \nu_i \prod_{1 \le j \ne i \le k} \frac{x - \mu_j}{\mu_i - \mu_j}$, and $L_i(\mu_i) = \nu_i, L_i(\mu_j) = 0 (i \ne j)$, *i.e.* $F(\mu_i) = \nu_i (i = 1, \ldots, t)$, so $F(x) = f(x)$ (by Lagrange interpolating polynomial theorem), and compute $a_0 = f(0)$.
(2) Compute $m \mid\mid \gamma = H_4(S, a_0) \oplus C$.
(3) Verify if $S = H_2(m, \gamma)P$ holds. If not, $\mathcal{DC}$ stops the process; otherwise, $\mathcal{DC}$ returns the plaintext m to $\mathcal{R}_i$.

5. Security analysis of the proposed CLMRThD scheme

Security is a very important property of cryptography scheme. In this section, we will define the security models and the security notions of the proposed CLMRThD scheme. The security notions are the indistinguishability of encryption under selective multi-ID chosen-ciphertext attacks (IND-sMID-CCA) and anonymity of receiver under selective multi-ID, chosen-ciphertext outsider attacks (Anon-sMID-CCOA). Then, we will prove that our CLMRThD scheme is provably secure in the robust security model by using the security theorems.

5.1. Security model

In order to deal with the security of the CLMRThD scheme, we will consider the adversarial model that determines the goal and the possible actions of the adversary. In our setting, the goals of the adversary are to distinguish two messages and receiver identity under a challenge ciphertext. In an attack, we assume that adversary may choose multiple identities, and already has a part of the receiver's private key by corrupting them. However, the number of private keys possessed by the adversary cannot exceed $t-1$. In the proposed CLMRThD, we define the following two kinds of adversaries:

Type I adversary $\mathcal{A}_1$: $\mathcal{A}_1$ is outside adversary who cannot access the master private key of KGC. But $\mathcal{A}_1$ can replace the user's public key with a value of his/her choice.

Type II adversary $\mathcal{A}_2$: $\mathcal{A}_2$ can get the master key. However does not allow him/her to replace public key of any user at any time.

Define the security of a CLMRThD scheme as a game played between an adversary $\mathcal{A} \in \{\mathcal{A}_1, \mathcal{A}_2\}$ and a challenger $\mathcal{C}$. During the game, $\mathcal{A}$ can make the following queries to $\mathcal{C}$.

$Create-User$ query: $\mathcal{C}$ generates private key and public key for the user U_i. $\mathcal{C}$ sends the user U_i's public key to $\mathcal{A}$.

$Public-Key-Retrieve$ query: $\mathcal{C}$ returns the matching user U_i's public key to $\mathcal{A}$.

$Public-Key-Replacement$ query: $\mathcal{C}$ replaces the associated user's public key with new public key chosen by himself/herself.

$Secret-Value-Extract$ query: $\mathcal{C}$ sends the user's secret value to $\mathcal{A}$.

$Partial-Private-Key-Extract$ query: $\mathcal{C}$ sends the user's partial private key to $\mathcal{A}$.

$Decryption-share$ query: $\mathcal{C}$ calculates decryption shares from received ciphertext using receivers' private key and sends them to $\mathcal{A}$.

Decryption query: $\mathcal{C}$ decrypts the received ciphertext and sends plaintext to $\mathcal{A}$.

We define the confidentiality of a CLMRThD scheme as the indistinguishability against selective multi-identity chosen ciphertext attack (IND-sMID-CCA). The IND-sMID-CCA game is defined as follows.

Game I In order to indicate the confidentiality of the CLMRThD scheme.

Initialization. $\mathcal{A}$ selects n receivers $R^* = \{R_1^*, R_2^*, \ldots, R_n^*\}$ with identity $ID^* = \{ID_1^*, ID_2^*, \ldots, ID_n^*\}$ respectively and sends ID^* to $\mathcal{C}$.

Note, here we can assume that $\mathcal{A}$ has already learned about the private keys of at most $t-1$ corrupted receivers among these target receivers (*e.g.* with identities $ID_1^*, ID_2^*, \ldots, ID_{t-1}^*$).

Setup. The challenger runs *setup* to generate *params* and *msk*. $\mathcal{C}$ sends *params* to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ could adaptively make oracle queries aforementioned, but do not allow him/her to make $Partial-private-Extract/Public-Key-Replace$ query with $ID \in \{ID_t^*, ID_{t+1}^*, \ldots, ID_n^*\}$ if he/she is $\mathcal{A}_1/\mathcal{A}_2$.

Challenge $\mathcal{A}$ selects two distinct plaintexts m_0 and m_1 with equal length, then sends $\{m_0, m_1\}$ to $\mathcal{C}$. $\mathcal{C}$ randomly selects $\lambda \in \{0,1\}$ and uses $\{ID_1^*, ID_2^*, \ldots, ID_n^*\}$ and corresponding public key to encrypt the message m_λ for generation the ciphertext CT^*. Then $\mathcal{C}$ sends CT^* to $\mathcal{A}$.

Phase 2. In this phase, $\mathcal{A}$ can make the same queries as he/she does **in Phase 1**. However, $\mathcal{A}$ cannot make *Decryption* query with CT^* and $\{ID_1^*, ID_2^*, \ldots, ID_n^*\}$, and he/she also cannot make decryption share query about $ID \in \{ID_t^*, ID_{t+1}^*, \ldots, ID_n^*\}$.

Guess Finally, $\mathcal{A}$ outputs a guess $\lambda' \in \{0,1\}$. We say that $\mathcal{A}$ wins the game if $\lambda' = \lambda$.

We call such $\mathcal{A}$ as IND-sMID-CCA adversary. The advantage of $\mathcal{A}$ against the CLMRThD scheme is defined as below:

$$Adv_{CLMRThD}^{IND-sMID-CCA}(\mathcal{A}) = \mid Pr[\lambda' = \lambda] - 1/2 \mid .$$

Definition 5.1. We say a CLAMRE scheme is IND-sMID-CCA secure if there exists no polynomial-time attacker that can win the IND-sMID-CCA game with non-negligible advantage.

The IND-sMID-CCA game model is shown in Figure 2.

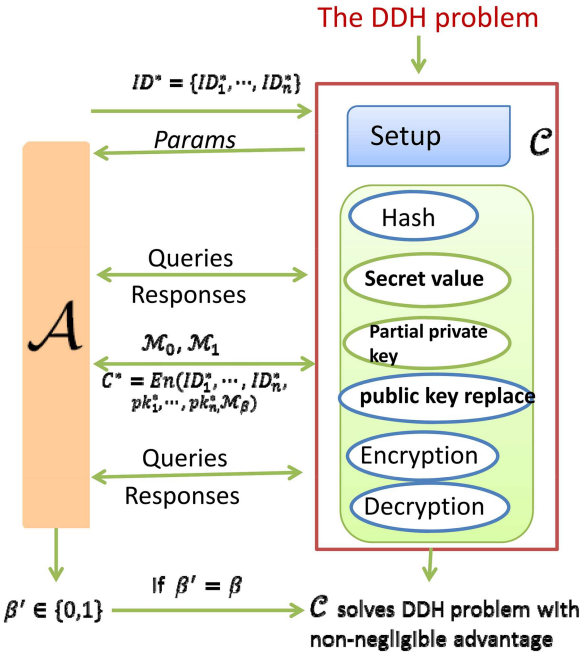

FIGURE 2. The indistinguishability of encryptions under IND-sMID-CCA game.

We define the receiver anonymity of a CLMRThD scheme as the anonymous indistinguishability against selective identity chosen ciphertext outsider attack (ANON-IND-sID-CCOA). The ANON-IND-sID-CCOA game is defined as bellow.

Game II In order to show the anonymity of the CLMRThD scheme.

Initialization. $\mathcal{A}$ selects two receivers $R^* = \{R_0^*, R_1^*\}$ with identity $ID^* = \{ID_0^*, ID_1^*\}$ respectively and sends them to $\mathcal{C}$.

Setup. The challenger runs *setup* to generate *params* and *msk*. $\mathcal{C}$ sends *params* to $\mathcal{A}$.

Phase 1. In this phase, $\mathcal{A}$ could adaptively make the oracle query aforementioned. However, he/she cannot make $Create-User, partial-key-Extract/Public-Key-Replace$ query with $ID \in ID^*$ if he/she is $\mathcal{A}_1/\mathcal{A}_2$.

Challenge. $\mathcal{A}$ submits a message m and a set of identities $\{ID_2, ID_3, \ldots, ID_n\}(n \geq t)$ to $\mathcal{C}$, $\mathcal{C}$ randomly selects $\lambda \in \{0,1\}$ and uses identities $ID_\lambda^*, ID_2, ID_3, \ldots, ID_n$ and corresponding public key to generate a ciphertext CT^*. Then $\mathcal{C}$ sends CT^* to $\mathcal{A}$.

Note, here we can also assume that $\mathcal{A}$ has already learned about the private keys of identities $ID_2, ID_3, \ldots, ID_t$.

Phase 2. In this phase, $\mathcal{A}$ can make the same queries as he/she does in Phase 1 except that he/she cannot make *Decryption* query with CT^* and $\Theta \subseteq \{ID_\lambda^*, ID_2, ID_3, \ldots, ID_n\}$, where $\mid \Theta \mid = t, ID_\lambda^* \in \Theta$, and he/she also cannot make decryption share query about $ID \in \{ID_0^*, ID_1^*\}$.

Guess Finally, $\mathcal{A}$ returns $\lambda' \in \{0,1\}$ as his/her guess value about λ. We say that $\mathcal{A}$ wins the game if $\lambda' = \lambda$. The advantage that $\mathcal{A}$ against the game is defined by $Adv_{CLAMRE}^{ANON-IND-sID-CCOA}(\mathcal{A}) = \mid Pr[\lambda' = \lambda] - 1/2 \mid$.

Definition 5.2. A CLMRThD scheme is said to be ANON-IND-sID-CCOA secure if $Adv_{CLAMRE}^{ANON-IND-sID-CCOA}(\mathcal{A})$ is negligible for any polynomial-time-bounded adversary $\mathcal{A}$.

The model of this game is shown in Figure 3.

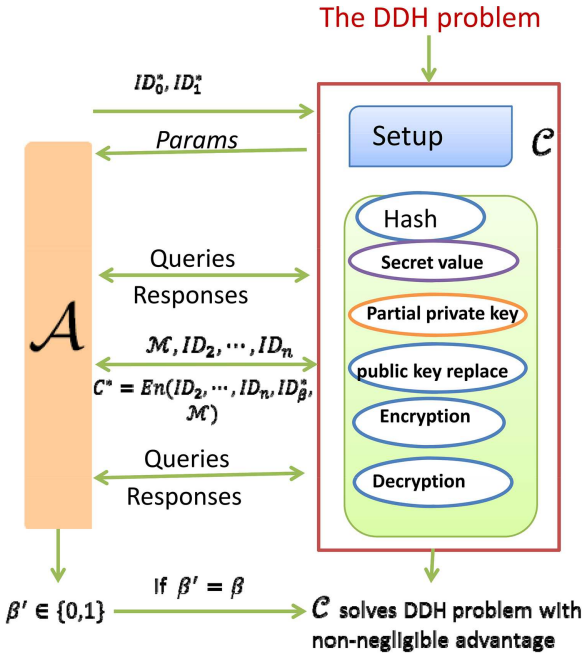

FIGURE 3. The anonymous (Anon)-sMID-CCA game.

5.2. Security theorems

In this subsection, we will discuss security of the proposed CLMRThD scheme. The analysis indicates that the proposed CLMRThD scheme is IND-sMID-CCA secure and ANON-IND-sID-CCOA secure against two types of adversaries $\mathcal{A}_1$, $\mathcal{A}_2$.

Theorem 5.3. *The proposed CLMRThD scheme correctly generates the ciphertext* $\sigma = (S, C, \nu_1, \ldots, \nu_n, \Gamma)$, *and at least* t *receivers act together can decrypt it appropriately.*

Proof. Receiver R_i can compute $(s_i + r_i)S = (r_i + t_i + k_i x)sP = s(r_i + t_i + k_i x)P = s(P_i + T_i + k_i P_{pub}) = U_i$ and $\mu_i = H_3(U_i, ID_i, pk_i)$. So, every receiver $\mathcal{R}_i$ can correctly compute decryption share μ_i using his/her private key $sk_i = (s_i, r_i)$.

A DC (or one of the receivers) collects t decryption shares $\mu_1, \mu_2, \ldots, \mu_t \in Z_q^*$ to construct a polynomial $F(x) = \sum_{i=1}^{t} L_i(x)$ of degree $t-1$ that passes through the t points $(\mu_1, \nu_1), (\mu_2, \nu_2), \ldots, (\mu_t, \nu_t)$, where $L_i(x) = \nu_i \prod_{1 \le j \ne i \le k} \frac{x - \mu_j}{\mu_i - \mu_j}$. So, $F(x) = f(x)$ (by Lagrange interpolating polynomial theorem), and compute $a_0 = f(0)$.

DC computes $m \mid\mid \gamma = H_4(S, a_0) \oplus C$ and $s = H_2(m, \gamma)$ is true. So, we proposed CLMRThD scheme is correct and consistent. □

Theorem 5.4. *In the random oracle model, our CLMRThD scheme is the IND-sMID-CCA secure against the adversary* $\mathcal{A}_1$ *with the hardness assumption of DDH problem.*

Proof. The basic idea of proof is contradiction. If proposed CLMRThD scheme is not IND-sMID-CCA secure, *i.e.* there exists a polynomial time-bounded adversary $\mathcal{A}_1$ that wins the game with non-negligible advantage. Then we can construct a probabilistic polynomial time-bounded challenger $\mathcal{C}$ to solve the DDH problem by interacting with the adversary $\mathcal{A}_1$, that is, for given an instance (P, aP, bP, X) of the DDH problem, the challenger $\mathcal{C}$ be able to determine if $X = ab \cdot P$ holds. The challenger $\mathcal{C}$ simulates the game for $\mathcal{A}_1$ as below.
The challenger $\mathcal{C}$ first maintains the following initial-empty lists in order to achieve the consistency between queries made by the adversary $\mathcal{A}_1$. □

Initialization. $\mathcal{A}_1$ selects n target identities $ID_1, \ldots, ID_n$ and sends them to $\mathcal{C}$.

Note, here we can assume that $\mathcal{A}_1$ has already learned about the private keys of at most $t-1$ corrupted receivers among these target receivers (*e.g.* with identities $ID_1, ID_2, \ldots, ID_{t-1}$).

Setup. $\mathcal{C}$ sets $P_{pub} \longleftarrow x \cdot P$, and executes setup algorithm to generate other parameters. Then $\mathcal{C}$ delivers $\{p, q, l_1, l_2, E, G, G_q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ to $\mathcal{A}_1$.

Phase 1. To simulate the random oracles, $\mathcal{C}$ maintains four lists $L_{H_i}^{list}$, where $L_{H_i}^{list}$ is initialized empty $(i = 1, 2, 3, 4)$. The four random oracles make following response to $\mathcal{A}_1's$ queries.

- $H_1(ID_i, P_i, T_i)$: $\mathcal{C}$ checks if (ID_i, P_i, T_i, k_i) exists in $L_{H_1}^{list}$. If so, $\mathcal{C}$ sends k_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly chooses a value $k_i \in Z_q^*$, inserts (ID_i, P_i, T_i, k_i) into $L_{H_1}^{list}$ and sends k_i.
- $H_2(m_i, \gamma_i)$: $\mathcal{C}$ checks if (m_i, γ_i, b_i) exists in $L_{H_1}^{list}$. If so, $\mathcal{C}$ returns s_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly chooses a value $b_i \in Z_q^*$, inserts (m_i, γ_i, b_i) into $L_{H_2}^{list}$ and returns b_i.
- $H_3(U_i, ID_i, pk_i)$: $\mathcal{C}$ checks if (U_i, ID_i, pk_i, μ_i) exists in $L_{H_3}^{list}$. If so, $\mathcal{C}$ returns μ_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly chooses a value $\mu_i \in Z_q^*$, inserts (U_i, ID_i, pk_i, μ_i) into $L_{H_3}^{list}$ and returns μ_i.
- $H_4(S_i, a_{i0})$: $\mathcal{C}$ checks if (S_i, a_{i0}, τ_i) exists in $L_{H_4}^{list}$. If so, $\mathcal{C}$ returns τ_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly selects an element $\tau_i \in \{0,1\}^{l_1+l_2}$, inserts (S_i, a_{i0}, τ_i) into $L_{H_4}^{list}$ and returns τ_i.

Phase 2. $\mathcal{A}_1$ can adaptively make queries to $\mathcal{C}$. $\mathcal{C}$ maintains a list L_R^{list}, which is initialized empty. Challenger $\mathcal{C}$ responses these queries made by adversary $\mathcal{A}_1$ as below.

- $Create - User(ID_{\mathcal{R}_i})$ query: $\mathcal{C}$ checks if $(ID_{\mathcal{R}_i}, t_i, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If so, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ executes the following processes.
- If $ID_{\mathcal{R}_i} \in \{ID_t, \ldots, ID_n\}$ holds, without losing generality, we suppose $ID_{\mathcal{R}_i} = ID_i (t \leq i \leq n)$, $\mathcal{C}$ randomly chooses $r_i, t_i \in Z_q^*$, computes $P_i = r_i P, T_i = t_i aP, k_i = H_1(ID_i, P_i, T_i)$ and sets $s_i \leftarrow \perp$. $\mathcal{C}$ inserts (ID_i, P_i, T_i, k_i) and $(ID_i, t_i, s_i, r_i, T_i, P_i)$ into $L_{H_1}^{list}$ and L_R^{list}, respectively. At last, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$.
- If $ID_{\mathcal{R}_i} \notin \{ID_1, \ldots, ID_n\}$, $\mathcal{C}$ randomly picks $r_i, s_i, k_i \in Z_q^*$, computes $T_i = s_i P - k_i P_{pub}, P_i = r_i P$. $\mathcal{C}$ inserts (ID_i, P_i, T_i, k_i) and $(ID_i, t_i, s_i, r_i, T_i, P_i)$ into $L_{H_1}^{list}$ and L_R^{list}, respectively. At last, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$.
- *Public-Key*-Retrieve(ID_{R_i}): $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$.
- *Public-key-Replacement* (ID_{R_i}, T_i', P_i'): $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ replaces (T_i, P_i) with (T_i', P_i'). Now challenger $\mathcal{C}$ sets $s_i = \perp, sk_i = (\perp, \perp)$.
- $Partial - Private - Key - Extract(ID_{R_i})$: $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns s_i to $\mathcal{A}_1$.
- $Secret - Value - Extract(ID_{R_i})$: $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns l_i to $\mathcal{A}_1$.
- $Decryption(ID_{R_1}, ID_{R_2}, \ldots, ID_{R_t}, \sigma_i)$: $\mathcal{C}$ checks if $\forall ID_{\mathcal{R}_j} \notin \{ID_1, \ldots, ID_n\} (j = 1, 2, \ldots, t)$ holds, where $\sigma_i = (S_i, C_i, \nu_{i1}, \ldots, \nu_{i1}, \Gamma)$. If yes, $\mathcal{C}$ looks for tuple $(ID_{R_j}, s_j, r_j, T_j, P_j)$ in L_R^{list} and uses (s_j, r_j) to compute decryption share μ_{ij}, and $\mathcal{DC}$ decrypts ciphertext σ_i using t shares collected. Otherwise there exists some $ID_{\mathcal{R}_i} \in \{ID_1, \ldots, ID_n\}$, $\mathcal{C}$ responds according to the following steps.
- $\mathcal{C}$ looks up $L_{H_4}^{list}$ for (S_i, a_{i0}, τ_i). If not, $\mathcal{C}$ outputs failure and stops.

– $\mathcal{C}$ searches the tuple (m_i, γ_i, s_i) from $L_{H_2}^{list}$, and checks if $S_i = s_i P$ holds. If so, $\mathcal{C}$ keeps (m_i, γ_i), if not, $\mathcal{C}$ outputs failure and stops.
– $\mathcal{C}$ checks if $C_i = \tau_i \oplus (m_i \parallel \gamma_i)$ holds. If not, $\mathcal{C}$ outputs failure and stops. Otherwise return m_i to the $\mathcal{A}_1$.

Challenge. After making above queries, $\mathcal{A}_1$ chooses two distinct plaintexts m_0 and m_1 with the length l_2 and sends them to challenger $\mathcal{C}, \mathcal{C}$ chooses $\lambda \in \{0, 1\}$ uniformly and executes the following steps.

– $\mathcal{C}$ sets $S^* \leftarrow b \cdot P$.
– Let $U_i = (k_i x + r_i)S^* + t_i X$, and computes $\mu_i = H_3(U_i, ID_i, pk_i), i = 1, 2, \ldots, n$.
– $\mathcal{C}$ chooses $a_0, a_1, \ldots, a_{t-1} \in_R Z_q^*$ to construct a polynomial $f(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1}$ of degree $t-1$, and computes $\nu_i^* = f(\mu_i)(i = 1, 2, \ldots, n)$.
– $\mathcal{C}$ chooses $\gamma^* \in \{0, 1\}^{l_2}$ at random and computes $\tau^* = H_4(S^*, a_0) \in \{0, 1\}^{l_1+1_2}$ and $C^* = \tau^* \oplus (m_\lambda \parallel \gamma^*)$.

Final, $\mathcal{C}$ sends the ciphertext $\sigma^* = (S^*, C^*, \nu_1^*, \nu_2^*, \ldots, \nu_n^*, \Gamma)$.

Phase 3. In this phase, $\mathcal{A}_1$ issues new queries similar to in Phase 1 and Phase 2 except that he cannot make decryption queries with $\Omega \subseteq \{ID_1, ID_2, \ldots, ID_n\}$ and σ^*, where $\mid \Omega \mid = t$. $\mathcal{A}_1$ is not allowed to make decryption share query if $ID_i \in \{ID_t, \ldots, ID_n\}$.

Guess. $\mathcal{A}_1$ outputs $\lambda' \in \{0, 1\}$ as guess value about λ. If $\lambda = \lambda'$, then $\mathcal{C}$ outputs 1, otherwise, $\mathcal{C}$ outputs 0. $\mathcal{A}_1$ wins the game if and only if $\lambda = \lambda'$ holds.

Based on the above oracle queries, the simulation of $\mathcal{C}$ is perfect. Next, we consider the probability that challenger $\mathcal{C}$ fails in Game I. Combined with the previous description, we know $\mathcal{C}$ fails in *Decryption* query if (S_i, a_{i0}) is not in $L_{H_4}^{list}$. The probability that $\mathcal{A}_1$ can correctly guess the output of $H_4 : \{0, 1\}^* \rightarrow \{0, 1\}^{l_1+l_2}$ is $1/2^{l_1+l_2}$. Therefore, the probability of $\mathcal{C}$ failure in the game I is less than $q_d/2^{l_1+l_2}$, where q_d denotes the Decryption query times in the game.

If $X = abP$ holds, then σ^* is valid ciphertext. Thus, $\mathcal{A}_1$ is able to distinguish λ with non-negligible advantage ϵ. $Pr[c = 1 \mid X = abP] = Pr[\lambda = \lambda' \mid X = abP] = \frac{1}{2} + \epsilon$.
If $X \neq abP$,then the ciphertext distribution is random and uniform when $\lambda = 0$ or $\lambda = 1$. So, $\mathcal{A}_1$ cannot distinguish λ with any advantage.
$Pr[c = 1 \mid X \neq abP] = Pr[\lambda = \lambda' \mid X \neq abP] = \frac{1}{2}$.

Probability analysis: Suppose that adversary $\mathcal{A}_1$ can break the IND-sMID-CCA security of the proposed CLMRThD scheme with non-negligible advantage ϵ, let's analyze Probability of challenger $\mathcal{C}$ solving the DDH problem. Assume that adversary $\mathcal{A}_1$ makes H_i query at most $q_i (i = 1, 2, 3, 4)$ times, q_c times create user queries, and q_d times threshold decryption queries. We also assume that challenger $\mathcal{C}$ successfully simulates the hash oracle $H_i (i = 1, 2, 3, 4)$ and never repeats H_i-oracle query with the same inputs.

(1) $\mathcal{C}$ succeeds *Create user* query with probability $(1 - \frac{q_1}{q})$ each time. So the execution of *Create user* oracle is successful q_c times with the probability

$$(1 - \frac{q_1}{q})^{q_c} \geq (1 - \frac{q_1 q_c}{q}).$$

(2) $H_i (i = 2, 3)$ hash query succeeds with probability $(1 - \frac{q_i}{q})$ each time. Therefore the execution of H_2 hash query is successful q_i times with the probability

$$(1 - \frac{q_i}{q})^{q_i} \geq (1 - \frac{q_i^2}{q}).$$

(3) Similarly, the execution of H_4 hash function succeeds q_4 times with the probability

$$(1 - \frac{q_4}{2^{l_1+l_2}})^{q_4} \geq (1 - \frac{q_4^2}{2^{l_1+l_2}}).$$

(4) As can be seen from the previous analysis, Decryption query succeeds with the probability $(1 - \frac{q_d}{2^{l_1+l_2}})$. Therefore, we know $\mathcal{C}$ can solve the DDH problem with a non-negligible advantage

$$\varepsilon' \geq (1 - \frac{q_1 q_c}{q})(1 - \frac{q_2^2}{q})(1 - \frac{q_3^2}{q})(1 - \frac{q_4^2}{2^{l_1+l_2}})(1 - \frac{q_d}{2^{l_1+l_2}})\varepsilon.$$

Because of the DDH problem is difficult. Therefore, the proposed CLMRThD scheme is IND-sMID-CCA secure against $\mathcal{A}_1$.

Theorem 5.5. *Our CLMRThD scheme is IND-sMID-CCA secure against Type II adversary $\mathcal{A}_2$ under random oracle model with the difficulties of Decision Diffie-Hellman problem.*

Proof. $\mathcal{A}_2$ is the polynomial time-bounded adversary, if $\mathcal{A}_2$ can break the security of the proposed CLMRThD scheme. Then we can construct a PPT challenger $\mathcal{C}$ to solve the DDH problem by interacting with the adversary $\mathcal{A}_2$; that is, for given an instance (P, aP, bP, X) of the DDH problem, the challenger $\mathcal{C}$ be able to decide if $X = ab \cdot P$ holds. The challenger $\mathcal{C}$ maintains the following initial-empty lists in order to achieve the consistency between queries made by the adversary $\mathcal{A}_2$: □

Initialization. $\mathcal{A}_2$ selects n target receivers $\mathcal{R}_1, \ldots, \mathcal{R}_n$, the $ID_1, \ldots, ID_n$ denote identity of $\mathcal{R}_1, \ldots, \mathcal{R}_n$, respectively.

Similarly, we also assume that $\mathcal{A}_2$ has already learned about the private keys of at most $t-1$ corrupted receivers among these target receivers (*e.g.* with identities $ID_1, ID_2, \ldots, ID_{t-1}$).

Setup $\mathcal{C}$ picks $x \in Z_q^*$ at random as system private key, and computes corresponding public key $P_{pub} = x \cdot P$. $\mathcal{C}$ performs *Setup* algorithm to construct other parameters. At last, $\mathcal{C}$ delivers $\{p, q, l_1, l_2, E, G, G_q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ to $\mathcal{A}_2$ and master private key x to $\mathcal{A}_2$.

Phase 1. In this phase, To simulate the random oracles, $\mathcal{C}$ maintains four lists $L_{H_i}^{list}$, where $L_{H_i}^{list}$ is initialized empty $(i = 1, 2, 3, 4)$. The four random oracles make following response to $\mathcal{A}_2's$ queries .

- $H_1(ID_i, P_i, T_i) : \mathcal{C}$ checks if (ID_i, P_i, T_i, k_i) exists in $L_{H_1}^{list}$. If so, $\mathcal{C}$ sends k_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly chooses an value $k_i \in Z_q^*$, inserts (ID_i, P_i, T_i, k_i) into $L_{H_1}^{list}$ and sends k_i.
- $H_2(m_i, \gamma_i)$: $\mathcal{C}$ checks if (m_i, γ_i, b_i) exists in $L_{H_1}^{list}$. If so,$\mathcal{C}$ returns b_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly chooses an value $s_i \in Z_q^*$, inserts (m_i, γ_i, b_i) into $L_{H_2}^{list}$ and returns b_i.
- $H_3(U_i, ID_i, pk_i)$: $\mathcal{C}$ checks if (U_i, ID_i, pk_i, μ_i) exists in $L_{H_3}^{list}$. If so, $\mathcal{C}$ returns μ_i to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ randomly chooses an value $\mu_i \in Z_q^*$, inserts (U_i, ID_i, pk_i, μ_i) into $L_{H_3}^{list}$ and returns μ_i.
- $H_4(S_i, a_{i0})$: $\mathcal{C}$ checks if (S_i, a_{i0}, τ_i) exists in $L_{H_4}^{list}$. If so,$\mathcal{C}$ returns τ_i to $\mathcal{A}_2$. Otherwise, $\mathcal{C}$ randomly selects an element $\tau_i \in \{0,1\}^{l_1+l_2}$, inserts (S_i, a_{i0}, τ_i) into $L_{H_4}^{list}$ and returns τ_i.

Phase 2. In this phase, $\mathcal{A}_2$ can adaptively make a lot of queries to $\mathcal{C}$. $\mathcal{C}$ maintains a list L_R^{list}, which is initialized empty. These queries are responded as below.

- $Create - User(ID_{\mathcal{R}_i})$query: $\mathcal{C}$ checks if $(ID_{\mathcal{R}_i}, t_i, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If so, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_2$. Otherwise, $\mathcal{C}$ performs the following steps.
- If $ID_{\mathcal{R}_i} \in \{ID_t, \ldots, ID_n\}$ holds, Without losing generality, we suppose $ID_{\mathcal{R}_i} = ID_i$,$\mathcal{C}$ randomly chooses $r_i, t_i \in Z_q^*$, calculates $P_i = l_i \cdot aP, T_i = t_i P, k_i = H_1(T_i, P_i, ID_i), s_i = t_i + k_i x \bmod q$, $\mathcal{C}$ inserts (ID_i, P_i, T_i, k_i) and $(ID_i, t_i, s_i, r_i, T_i, P_i)$ into L_1^{list} and L_R^{list} respectively. At last, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_2$.
- If $ID_{\mathcal{R}_i} \notin \{ID_1, \ldots, ID_n\}$,$\mathcal{C}$ randomly picks $t_i, r_i, k_i \in Z_q^*$, computes $P_i = r_i \cdot P, T_i = t_i P, s_i = t_i + k_i x \bmod q$. $\mathcal{C}$ inserts (ID_i, P_i, T_i, k_i) and $(ID_i, t_i, s_i, r_i, T_i, P_i)$ into L_1^{list} and L_R^{list} respectively. At last, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_2$.
- *Public-Key*-Retrieve$(ID_{R_i}) : \mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_2$.

- *Public-key-Replacement* (ID_{R_i}, T_i', P_i'): $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ replaces (T_i, P_i) with (T_i', P_i'). Now challenger $\mathcal{C}$ sets $s_i = \perp, sk_i = (\perp, \perp)$.
- $Partial - Private - Key - Extract(ID_{R_i}) : \mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns s_i to $\mathcal{A}_2$.
- $Secret - Value - Extract(ID_{R_i}) : \mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns l_i to $\mathcal{A}_2$.
- $Decryption(ID_{R_1}, ID_{R_2}, \ldots, ID_{R_t}, \sigma_i)$: $\mathcal{C}$ checks if $\forall ID_{\mathcal{R}_j} \notin \{ID_1, \ldots, ID_n\}(j = 1, 2, \ldots, t)$ holds, where $\sigma_i = (S_i, C_i, \nu_{i1}, \ldots, \nu_{in}, \Gamma)$. If yes, $\mathcal{C}$ looks up L_R^{list} for $(ID_{R_j}, s_j, r_j, T_j, P_j)$ and uses (s_j, r_j) to compute decryption share μ_{ij}, and $\mathcal{DC}$ decrypts ciphertext σ_i using t shares collected . Otherwise there exists some $ID_{\mathcal{R}_i} \in \{ID_1, \ldots, ID_n\}$,$\mathcal{C}$ responds according to the following steps.
- $\mathcal{C}$ looks up $L_{H_4}^{list}$ for (S_i, a_{i0}, τ_i). If not,$\mathcal{C}$ output failure and stops.
- $\mathcal{C}$ search the tuple (m_i, γ_i, s_i) from $L_{H_2}^{list}$, and check if $S_i = s_i P$ holds. If so, $\mathcal{C}$ keeps (m_i, ω_i) , If not, $\mathcal{C}$ outputs failure and stops.
- -$\mathcal{C}$ checks if $C_i = \tau_i \oplus (m_i \parallel \gamma_i)$ holds . If not, $\mathcal{C}$ output failure and stops. Otherwise return m_i to the $\mathcal{A}_2$

Challenge. After making above queries, $\mathcal{A}_2$ chooses two distinct plaintexts m_0 and m_1 with the length l_2 and sends them to challenger $\mathcal{C}$,$\mathcal{C}$ chooses $\lambda \in \{0, 1\}$ uniformly and executes the following steps.

- $\mathcal{C}$ sets $S^* \leftarrow b \cdot P$.
- Let $U_i = (k_i x + t_i)S^* + r_i \cdot X$, and computes $\mu_i = H_3(U_i, ID_i, pk_i), i = 1, 2, \ldots, n$
- $\mathcal{C}$ chooses $a_0, a_1, \ldots, a_{t-1} \in_R Z_q^*$ to construct a polynomial $f(x) = a_0 + a_1 x + \cdots + a_{t-1}x^{t-1}$ of degree $t-1$, and computes $\nu_i^* = f(\mu_i)(i = 1, 2, \ldots, n)$
- $\mathcal{C}$ chooses $\gamma^* \in \{0, 1\}^{l_2}$ at random, and computes $\tau^* = H_4(S^*, a_0) \in \{0, 1\}^{l_1+1_2}$ and $C^* = \tau^* \oplus (m_\lambda \parallel \gamma^*)$. Final, $\mathcal{C}$ sends the ciphertext $\sigma^* = (S^*, C^*, \nu_1^*, \nu_2^*, \ldots, \nu_n^*, \Gamma)$.

Phase 3. In this phase, $\mathcal{A}_2$ can make the same queries in Phase 1 and Phase 2 except that he cannot make Decryption queries with $\Omega \subseteq \{ID_1, ID_2, \ldots, ID_n\}$ and σ^*,where $\mid \Omega \mid = t$.$\mathcal{A}_2$ is not allowed to make H_3 decryption share query if $ID_i \in \{ID_t, \ldots, ID_n\}$

guess $\mathcal{A}_2$ outputs $\lambda' \in \{0, 1\}$ as his/her guess value about λ. If $\lambda = \lambda'$, then $\mathcal{C}$ outputs 1,otherwise,$\mathcal{C}$ outputs 0.$\mathcal{A}_2$ wins the game if and only if $\lambda = \lambda'$ holds.

According to above oracle queries, we know the simulation of $\mathcal{C}$ is perfect. Now, we analyze the probability that $\mathcal{C}$ fails in Game I. Based on the above description, we know $\mathcal{C}$ fails in Decryption query if (S_i, e_i) is not in L_4^{list}. The probability that A1 can correctly guess the output of $H_4 : \{0, 1\}^* \rightarrow \{0, 1\}^{l_1+l_2}$ is $1/2^{l_1+l_2}$. Therefore, the probability that $\mathcal{C}$ fails in the game is less than $q_d/2^{l_1+l_2}$, where q_d denotes the Decryption queries involved in the game.

If $X = abP$ holds, then σ^* is valid ciphertext. Thus, $\mathcal{A}_1$ is able to distinguish λ with non-negligible advantage ϵ.

$Pr[c = 1 \mid X = abP] = Pr[\lambda = \lambda' \mid X = abP] = \frac{1}{2} + \epsilon$.

If $X \neq abP$, then the ciphertext distribution is random and uniform when $\lambda = 0$ *or* $\lambda = 1$. So, $\mathcal{A}_1$ cannot distinguish λ with any advantage.

$Pr[c = 1 \mid X \neq abP] = Pr[\lambda = \lambda' \mid X \neq abP] = \frac{1}{2}$.

Therefore, if $\mathcal{A}_2$ can break the IND-sMID-CCA security of the proposed CLMRThD scheme with non-negligible advantage ϵ. According to Theorem 5.4, we know $\mathcal{C}$ can solve the DDH problem with a non-negligible advantage $\varepsilon' \geq (1 - \frac{q_1 q_c}{q})(1 - \frac{q_2^2}{q})(1 - \frac{q_3^2}{q})(1 - \frac{q_4^2}{2^{l_1+l_2}})(1 - \frac{q_d}{2^{l_1+l_2}})\varepsilon$. Due to the DDH problem is hard. Therefore, the proposed CLMRThD scheme is IND-sMID-CCA secure against adversary $\mathcal{A}_2$.

Theorem 5.6. *In the random oracle model, our proposed CLMRThD scheme is ANON-IND-sID-CCOA secure against the adversary* $\mathcal{A}_1$ *with the difficulty assumption of DDH problem.*

Proof. Assume that the adversary $\mathcal{A}_1$ can breach our CLMRThD scheme, then we will be able to design a challenger $\mathcal{C}$ for solving an instance of DDH problem, that is, for given an instance $(P, a \cdot P, b \cdot P, X)$ of DDH problem, the challenger $\mathcal{C}$ can determine if $X = abP$ holds by interacting with the adversary $\mathcal{A}_1$. Similar to Theorem 5.4, let the lists $L_{H_i}^{list}(i = 1, 2, 3, 4)$ and L_R^{list} are maintained by the challenger $\mathcal{C}$. □

Initialization. The adversary $\mathcal{A}_1$ selects two target users $\mathcal{R}_0^*, \mathcal{R}_1^*$ with identity ID_0^*, ID_1^*, respectively.

Setup: $\mathcal{C}$ sets $P_{pub} \longleftarrow x \cdot P$, and implements *Setup* algorithm to construct other parameters. At last, $\mathcal{C}$ delivers $\{p, q, l_1, l_2, E, G, G_q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ to $\mathcal{A}_1$.

Phase 1. The challenger $\mathcal{C}$ responses to the adversary $\mathcal{A}_1$'s queries in the following ways:

Hash queries to $H_i(i = 1, 2, 3, 4)$: these queries are the same as performed in Theorem 5.4.

Phase 2. Now, the challenger $\mathcal{C}$ will respond to the queries made by the adversary $\mathcal{A}_1$ in the following ways:

- $Create - User(ID_{\mathcal{R}_i})$ query: $\mathcal{C}$ checks if $(ID_{\mathcal{R}_i}, r_i, s_i, t_i, T_i, P_i)$ exists in L_R^{list}. If so, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$. Otherwise, $\mathcal{C}$ executes the below processes.
- If $ID_{\mathcal{R}_i} = ID_j^*$ for $j \in \{0, 1\}$ holds, $\mathcal{C}$ randomly chooses $r_i, t_i \in Z_q^*$, computes $T_i = t_i \cdot aP, P_i = r_i \cdot P, k_i = H_1(ID_i, P_i, T_i)$ and sets $s_i \leftarrow \perp$. $\mathcal{C}$ inserts (ID_i, P_i, T_i, k_i) and $(ID_i, t_i, s_i, r_i, T_i, P_i)$ into $L_{H_1}^{list}$ and L_R^{list}, respectively. At last, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$.
- Otherwise $ID_{\mathcal{R}_i} \notin \{ID_0, ID_1\}$, $\mathcal{C}$ randomly picks $r_i, s_i, k_i \in Z_q^*$, computes $T_i = s_i P - k_i P_{pub}, P_i = r_i P$. $\mathcal{C}$ inserts (ID_i, P_i, T_i, k_i) and $(ID_i, t_i, s_i, r_i, T_i, P_i)$ into L_1^{list} and L_R^{list}, respectively. At last, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$.
- $Public - Key - Retrieve(ID_{R_i})$: $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_i first. Then, $\mathcal{C}$ returns (T_i, P_i) to $\mathcal{A}_1$.
- $Public - key - Replace(ID_{R_i}, T_i', P_i')$: $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ replaces (T_i, P_i) with (ID_i, T_i', P_i').
- $Secret - Value - Extract(ID_{R_i})$: $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns l_i to $\mathcal{A}_1$.
- $Partial - Private - Key - Extract(ID_{R_i})$: $\mathcal{C}$ checks if $(ID_{R_i}, s_i, r_i, T_i, P_i)$ exists in L_R^{list}. If not, $\mathcal{C}$ makes the Create-User query with ID_{R_i} first. Then, $\mathcal{C}$ returns s_i to $\mathcal{A}_1$.
- $Decryption(ID_{R_1}, ID_{R_2}, \ldots, ID_{R_t}, \sigma_i)$: $\mathcal{C}$ checks if $\forall ID_{\mathcal{R}_i} \notin \{ID_0^*, ID_1^*\}$ holds, where $\sigma_i = (S_i, C_i, \nu_{i1}, \ldots, \nu_{in}, \Gamma)$. If yes, $\mathcal{C}$ looks up L_R^{list} for $(ID_{R_j}, s_j, r_j, T_j, P_j)$ and uses (s_j, r_j) to compute decryption share μ_{ij}, and $\mathcal{DC}$ decrypts ciphertext σ_i using t shares collected. Otherwise there exists a $ID_{\mathcal{R}_i} \in \{ID_0^*, ID_1^*\}$, $\mathcal{C}$ responds according to the following steps.
- $\mathcal{C}$ looks up $L_{H_4}^{list}$ for (S_i, a_{i0}, τ_i). If not, $\mathcal{C}$ outputs failure and stops.
- $\mathcal{C}$ searches the tuple (m_i, γ_i, s_i) from $L_{H_2}^{list}$, and checks if $S_i = s_i P$ holds. If so, $\mathcal{C}$ keeps (m_i, ω_i), if not, $\mathcal{C}$ outputs failure and stops.
- $\mathcal{C}$ checks if $C_i = \tau_i \oplus (m_i \parallel \gamma_i)$ holds. If not, $\mathcal{C}$ outputs failure and stops. Otherwise return m_i to the $\mathcal{A}_1$.

Challenge. After making above queries, $\mathcal{A}_1$ picks plaintext m together with identities $\{ID_2, \ldots, ID_n\}(n \geq t)$ on which he wants to be challenged. $\mathcal{C}$ chooses $\lambda \in \{0, 1\}$ at random and implements the following process.

Here, we also assume that $\mathcal{A}_1$ has already learned about the private keys of at most $t - 1$ corrupted receivers among these target receivers (*e.g.* with identities $ID_2, ID_3, \ldots, ID_t$).

- $\mathcal{C}$ sets $S^* \leftarrow b \cdot P$.
- Let $U_i = (k_i x + r_i)S^* + t_i X$, and computes $\mu_i = H_3(U_i, V_i, ID_i, pk_i), i = 1, 2, \ldots, n$.
- $\mathcal{C}$ chooses $a_0, a_1, \ldots, a_{t-1} \in_R Z_q^*$ to construct a polynomial $f(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1}$ of degree $t - 1$, and computes $\nu_i^* = f(\mu_i)(i = 1, 2, \ldots, n)$.
- $\mathcal{C}$ chooses $\gamma^* \in \{0, 1\}^{l_2}$ at random, and computes $\tau^* = H_4(S^*, a_0) \in \{0, 1\}^{l_1 + 1_2}$ and $C^* = \tau^* \oplus (m_\lambda \parallel \gamma^*)$. Final, $\mathcal{C}$ sends the ciphertext $\sigma^* = (S^*, C^*, \nu_1^*, \nu_2^*, \ldots, \nu_n^*, \Gamma)$.

Phase 3. In this phase, $\mathcal{A}_1$ can make the same queries in phase 1 and Phase 2 except that he cannot make Decryption queries with $\Omega \subseteq \{ID_\lambda, ID_2, \ldots, ID_n\}$ and σ^*, where $\mid \Omega \mid = t, ID_\lambda \in \Omega$, $\mathcal{A}_1$ also cannot make decryption share queries with $ID \in \{ID_\lambda, ID_{t+1}, \ldots, ID_n\}$.

TABLE 1. The notation and definition of different time complexities.

Notation	Definition and conversion
T_m	The runtime of executing a modular multiplication in Z_q^*
T_e	The runtime of executing a modular exponentiation in $Z_q^*, T_e \approx 240T_m$
T_{bp}	The runtime of a bilinear pairing operation, $T_{bp} \approx 5T_e$
T_h	The runtime of executing a hash operation, $T_h \approx 23T_m$
T_{sm}	The runtime of executing an elliptic curve scalar point multiplication operation or an exponentiation operation in a multiplicative group, $T_{sm} = 29T_m$
T_a	The cost of an addition in an additive group or a multiplication in a multiplicative group, $T_a = 0.12T_m$

Guess. $\mathcal{A}_1$ outputs $\lambda' \in \{0,1\}$ as his/her guess value about λ. If $\lambda = \lambda'$, then $\mathcal{C}$ outputs 1, otherwise, $\mathcal{C}$ outputs 0. $\mathcal{A}_1$ wins the game if and only if $\lambda = \lambda'$ holds.

Based on the above oracle queries, the simulation of $\mathcal{C}$ is perfect. Next, we consider the probability that challenger $\mathcal{C}$ fails in Game I. Combined with the previous description, we know $\mathcal{C}$ fails in *Decryption* query if (S_i, e_i) is not in $L_{H_4}^{list}$. The probability that $\mathcal{A}_1$ can correctly guess the output of $H_4 : \{0,1\}^* \to \{0,1\}^{l_1+l_2}$ is $1/2^{l_1+l_2}$. Therefore, the probability of $\mathcal{C}$ failure in the Game I is less than $q_d/2^{l_1+l_2}$, where q_d denotes the Decryption query times in the game.

If $X = abP$ holds, then σ^* is valid ciphertext. Thus, $\mathcal{A}_1$ is able to distinguish λ with non-negligible advantage ϵ. $Pr[c = 1 \mid X = abP] = Pr[\lambda = \lambda' \mid X = abP] = \frac{1}{2} + \epsilon$.

If $X \neq abP$, then the ciphertext distribution is random and uniform when $\lambda = 0$ or $\lambda = 1$. So, $\mathcal{A}_1$ cannot distinguish λ with any advantage.

$Pr[c = 1 \mid X \neq abP] = Pr[\lambda = \lambda' \mid X \neq abP] = \frac{1}{2}$.

Therefore, if $\mathcal{A}_1$ can break the ANON-IND-sMID-CCOA security of the proposed CLMRThD scheme with non-negligible advantage ϵ, then according to Theorem 5.4, challenger $\mathcal{C}$ can solve the DDH problem with a non-negligible advantage $\varepsilon' \geq (1 - \frac{q_1 q_c}{q})(1 - \frac{q_2^2}{q})(1 - \frac{q_3^2}{q})(1 - \frac{q_4^2}{2^{l_1+l_2}})(1 - \frac{q_d}{2^{l_1+l_2}})\varepsilon$. Because of the DDH problem is difficult. Therefore, the proposed CLMRThD scheme is ANON-IND-sMID-CCOA secure against $\mathcal{A}_1$.

Theorem 5.7. *The proposed CLMRThD scheme is ANON-IND-sID-CCOA secure against the adversary $\mathcal{A}_2$ with the difficulty of DDH problem under the random oracle model.*

Proof. In here, to save space, we will not give the details. Specific ideas of proof can reference Theorem 5.5. □

6. PERFORMANCE ANALYSIS

In this section, we mainly analyzed computational cost of the proposed CLMRThD scheme. The proposed CLMRThD scheme compare with [5, 14, 15, 18, 29] in performance. Let G_1 be an additive group defined on a super singular elliptic curve over a prime field F_p with the prime order q, and the length of q and p are 512 bits and 160 bits, respectively. The Tate bilinear pairing $\hat{e} : G_1 \times G_1 \to G_2$. According to [7, 16, 20, 28], we give the concepts and conversion relations for different time complexity, see Table 1.

We denote n and t the number of the receivers and threshold value, respectively. In order to encrypt a given message m, in [5]'s scheme, the sender needs to perform $2n + 4$ scale multiplication operations in G_1, one bilinear pairing operations, n hash operations and $n + 1$ addition operations in an additive group. Therefore, the computation cost of the sender is $T_{bp} + (2n + 4) \times T_{sm} + (n + 1) \times T_a + n \times T_h \approx (81n + 1287)T_m$. For decrypting the received ciphertext, t receivers need to implement the following operations: t scale multiplication in G_1, $2t$ bilinear pairings, t addition operations in an additive group. Therefore, the cost of decryption in the scheme of [5] is $2t \times T_{bp} + t \times T_{sm} + t \times T_a \approx 2429tT_m$. Computation cost of scheme of [18] is closer to scheme of [5], its encryption and decryption costs are $(81n + 1310)T_m$ and $(2429t + 23)T_m$, respectively. The efficient of scheme of [29] is better than scheme of [5, 18], and the runtime of encryption and decryption are $(81n + 104)T_m$ and $(1229t + 1298)T_m$. In schemes of [14, 15], only need to operate: 5 scale multiplication, 1 bilinear pairing,

3 hashs, the cost of encryption is $1931T_m$. However, the two schemes take a lot of running time and expensive communication costs to establish key shares, and operating costs up to $1229n$, and the runtime required for decryption is very high, reaching $(29t^2 + 7502t + 23)T_m$.

In proposed CLMRThD scheme, to encrypt a given message m, the sender needs to perform below operations: n times addition in G_1, $2n + 1$ times scale multiplication in G_1, $n + 2$ times hash. Therefore, in our scheme, the cost of encryption is $(2n + 1) \times T_{sm} + n \times T_a + (n + 2) \times T_h \approx (81n + 75)T_m$. In order to get plaintext from the received ciphertext, a total of t receivers need to perform $2t$ scale multiplication operations, $t + 2$ hash operations. The cost of decryption in our scheme is $2t \times T_{sm} + (t + 2) \times T_h \approx (81t + 46)T_m$. According to comparisons in Table 2, we can conclude that the proposed CLMRThD scheme has much less runtime in both encryption and decryption than the recent schemes. Therefore, our proposed CLMRThD scheme has better performance.

About the security comparison of schemes, shown in Table 2, scheme [18] is not secure. Scheme [5, 14, 15, 18] did not provide the proof for anonymity, but schemes [14, 15] are CPA and CCA secure in confidentiality, and scheme [29] is CCA secure in both confidentiality and anonymity. Our scheme can achieve the CCA security in both confidentiality and anonymity under the random oracle model against the adversaries defined in CL-PKC system.

Table 2. Performance comparison.

	Encryption cost	Decryption cost	Set key share	Ciphertext	Secrity condition	Anonymity
[18]	$(2n+3)T_{sm} + T_{bp} + nT_a$ $+(n+1)T_h \approx (81n + 1310)T_m$	$2tT_{bp} + t(T_{sm} + T_a) + T_h$ $\approx (2429t + 23)T_m$		$(n+3)v_1$ $+\|M\|$	No	–
[29]	$(2n+2)T_{sm} + nT_a$ $+(n+2)T_h \approx (81n + 104)T_m$	$(t+1)(T_{bp} + T_{sm} + T_a)$ $+3T_h \approx (1229t + 1298)T_m$		$(n+1)v_1$ $+\|M\|$	CCA	Yes
[5]	$(2n+4)T_{sm} + (n+1)T_a + nT_h$ $+T_{bp} \approx (81n + 1287)T_m$	$2tT_{bp} + tT_{sm} + tT_a$ $\approx 2429tT_m$		$(n+1)v_1$ $+v_2$	CPA	–
[15]	$5T_{sm} + T_{bp} + 3T_h$ $\approx 1931T_m$	$6tT_{bp} + t(t+7)T_{sm}$ $+(2t+3)T_a + (4t+1)T_h$ $\approx (29t^2 + 7502t + 23)T_m$	$nT_{bp} + nT_{sm}$ $\approx 1229nT_m$	$3v_1 + 2v_q$	CPA CCA	–
[14]	$5T_{sm} + T_{bp} + 3T_h$ $\approx 1931T_m$	$6tT_{bp} + t(t+7)T_{sm}$ $+(2t+3)T_a + (4t+1)T_h$ $\approx (29t^2 + 7502t + 23)T_m$	$nT_{bp} + nT_{sm}$ $\approx 1229nT_m$	$3v_1 + 2v_q$	CPA CCA	–
Our	$(2n+1)T_{sm} + (n+2)T_h$ $+nT_a \approx (81n + 75)T_m$	$2tT_{sm} + (t+2)T_h$ $\approx (81t + 46)T_m$		$v_1 + \| M \|$ $+nv_q$	CCA	Yes

v_1, the bit length of an element in G_1; v_2, the bit length of an element in G_2; v_q, the bit length of an element in Z_q^*; t, the threshold value of threshold decryption scheme; n, the number of receivers; $\| M \|$, the bit length of plaintext; –, the author did not consider in the article.

7. Conclusion

In this study, we proposed an efficient CLMRThD scheme using the ECC. Performance analysis shows that our scheme has much less runtime than existing scheme. We also indicate that the proposed scheme is CCA secure in both confidentiality and anonymity under the random oracle model with the difficulties of decision Diffie–Hellman problem and against the adversaries defined in CL-PKC system.

In summary, our scheme has the following merits: (1) bilinear pairings are not used in encryption and

decryption process, (2) resisting all known security attacks, (3) achieving in both confidentiality of message and anonymity of receiver, (4) solving the problems of private key escrow problem and public key certificate management, (5) low computation and communication costs.

Acknowledgements. This research is supported by the National Natural Science Foundation of China under Grant No.61562012, the Innovation Group Major Research Projects of Department of Education of Guizhou Province under Grant No.KY[2016]026.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] S.S. Al-Riyami and K.G. Paterson, Certificateless public key cryptography, in *Proc. of the Ninth International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan* (2003) 452–473.

[2] D. Boneh and M. Franklin, Identity-based encryption from the Weil pairing, in Proceeding of Advances in Cryptology – CRYPTO 2001, edited by J. Kilian. Vol. 2139 of *Lecture Notes in Computer Science*. Springer-Verlag, New York (2001) 213–229.

[3] Z. Chai, Z. Cao and R. Lu, ID-based threshold decryption without random oracles and its application in key escrow, in *Proc. of the 3rd International Conference on Information Security*. ACM International Conference Proceeding Series (2004).

[4] Z. Chai, Z. Cao and Y. Zhou, Effient ID-based broadcast threshold decryption in ad hoc network, in *First International Multi-symposiums on Computer and Computational Sciences (IMSCCS'06), Hangzhou, China, June 20–24* (2006).

[5] Z. Chai, Z. Cao and X. Cao, Efficient ID-based multi-receiver threshold decryption. *Int. J. Found. Comput. Sci.* **18** (2007) 987–1004.

[6] L.Z. Deng, J.W. Zeng and X. Wang, An improved certificateless encryption scheme for telecare medicine information systems. *J. Internet Technol.* **18** (2017) 223–227.

[7] C.-I. Fan and Y.-F. Tseng, Anonymous multi-receiver identity-based authenticated encryption with CCA security. *Symmetry* **7** (2015) 1856–1881.

[8] D. He, S. Zeadally, N. Kumar and W. Wu, Efficient and anonymous mobile user authentication protocol using self-certified public key cryptography for multi-server architectures. *IEEE Trans. Inf. Foren. Secur.* **11** (2016) 2052–2064.

[9] J. Hong, J. Kim, J. Kim *et al.*, Fair threshold decryption with semi-trusted third parties, in ACISP 2009. Vol. 5594 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, Heidelberg (2009) 309–326.

[10] H.S. Ju, D.Y. Kim and D.H. Lee, Modified ID-based threshold decryption and its application to mediated ID-based encryption, in APWeb 2006. Vol. 3841 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin Heidelberg (2006) 720–725.

[11] K. Kim, J.H. Park and D.H. Lee, Selectively chosen ciphertext security in threshold public-key encryption. *Secur. Commun. Netw.* **9** (2016) 189–200.

[12] B. Libert and J.J. Quisquater, Efficient revocation and threshold pairing based cryptosystems, in *Proc. of the Twenty-second Annual Symposium on Principles of Distributed Computing-PODC'03*. ACM, Boston, MA (2003) 163–171.

[13] S. Liu and K. Chen, Identity-based threshold decryption revisited, in ISPEC'07. Vol. 4464 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin (2007) 329–343.

[14] Y. Long and K. Chen, Certificateless threshold cryptosystem secure against chosen-ciphertext attack. *Inform. Sci.* **177** (2007) 5620–5637.

[15] Y. Long, K. Chen and S. Liu, ID-based threshold decryption secure against adaptive chosen-ciphertext attack. *Comput. Electr. Eng.* **33** (2007) 166–176.

[16] A.J. Menezes, S.A. Vanstone and P.C. Van Oorschot, Handbook of Applied Cryptography. CRC Press, Boca Raton, FL (2001).

[17] Y. Ming and Y. Wang, Identity-based threshold decryption scheme without random oracle. *Chin. J. Electron.* **20** (2011) 323–328.

[18] H. Qin, Y. Dai and Z. Wang, Identity-based multi-receiver threshold signcryption scheme. *Secur. Commun. Netw.* **4** (2011) 1331–1337.

[19] H. Qin, X. Zhu and Y. Dai, Provably secure identity-based threshold decryption on access structure, in *10th International Conference on Computational Intelligence and Security CIS 2014, Kunming, China, November 15–16* (2014) 464–468.

[20] M. Scott, Implementing cryptographic pairings, in *Proc. of the Pairing-Based Cryptography, Tokyo, Japan, 2–4 July* (2007) 177–196.

[21] A. Shamir, Identity-based cryptosystems and signature schemes, in CRYPTO'84. Vol. 196 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin (1985) 47–53.

[22] V. Shoup and R. Gennaro, Securing threshold cryptosystems against chosen ciphertext attack, in EUROCRYPT'98. Vol. 1430 of *Lecture Notes in Computer Science*. Springer- Verlag, Berlin (1998) 1–16.

[23] X. Wang and Z. Chai, A provable security scheme of ID-based threshold decryption, in *Social Informatics and Telecommunications Engineering 2009*. IEEE Press, Adelaide (2009) 122–129.

[24] B. Waters, Efficient identity-based encryption without random oracles, in Advances in Cryptology – EUROCRYPT 2005, edited by R. Cramer. Vol. 3494 *Lecture Notes in Computer Science*. Springer-Verlag, Berlin (2005) 114–127.
[25] F. Xu, X. Lv, L.K. Jia, A new verifiable threshold decryption scheme without trusted center. *Intell. Autom. Soft Comput.* **17** (2011) 551–558.
[26] B. Yang, Y. Yu, F. Li and Y. Sun, Provably secure identity-based threshold unsigncryption scheme, in ATC'07. Vol. 4610 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin (2007) 114–122.
[27] G. Zhang, Certificateless threshold decryption scheme secure in the standard model, in Vol. 2 of *2nd IEEE International Conference on Computer Science and Information Technology, Beijing, China, August 08–11* (2009) 414–418.
[28] Y. Zhang, W. Liu, W. Lou and Y. Fang, Securing mobile ad hoc networks with certificateless public keys. *IEEE Trans. Dependable Secur. Comput.* **3** (2006) 386–399.
[29] M. Zhang, B. Yang and T. Takagi, Reconciling and improving of multi-receiver signcryption protocols with threshold decryption. *Secur. Commun. Netw.* **5** (2012) 1430–1440.