SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks 2012; **5**:439–449
Published online 4 May 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.326

RESEARCH ARTICLE

Anonymous proxy re-encryption

Jun Shao[1]*, Peng Liu[2], Guiyi Wei[1] and Yun Ling[1]

[1] College of Computer and Information Engineering, Zhejiang Gongshang University, 18 Xuezheng Street, Hangzhou, 310018 Zhejiang Province, China
[2] College of Information Sciences and Technology, Pennsylvania State University, University Park, PA 16801, U.S.A.

ABSTRACT

Proxy re-encryption (PRE) is a public key encryption that allows a semi-trusted proxy with some information (a.k.a., re-encryption key) to transform a ciphertext under one public key into another ciphertext under another public key. Because of this special property, PRE has many applications, such as the distributed file system. Some of these applications demand that the underlying PRE scheme is anonymous under chosen-ciphertext attacks (CCAs); that is, the adversary cannot identify the recipient of the original/transformed ciphertext, even if it knows the PRE key and can launch the CCA. However, to the best of our knowledge, *none* of the existing PRE schemes satisfy this requirement. In this work, we propose the *first* anonymous PRE with CCA security and collusion resistance. Our proposal is proved in the *random oracle model* based on the DDH assumption. Copyright © 2011 John Wiley & Sons, Ltd.

KEYWORDS

proxy re-encryption; chosen-ciphertext security; collusion resistance; anonymity; random oracle model

***Correspondence**

Jun Shao, College of Computer and Information Engineering, Zhejiang Gongshang University, Room 420, SCIE Building, 18 Xuezheng Street, Hangzhou, 310018 Zhejiang Province, China.
E-mail: chn.junshao@gmail.com

1. INTRODUCTION

There are many applications, such as the distributed file system [1,2], demanding that the ciphertext under one public key can be transformed into another ciphertext of the same message under another public key. However, it is not an easy job if the message is not allowed to be revealed during the transformation. To solve the problem, Blaze *et al.* [3] proposed the concept of proxy re-encryption (PRE), where a semi-trusted proxy with some information (a.k.a., re-encryption key) can re-encrypt a ciphertext for the delegator into a ciphertext of the same message for the delegatee. However, the proxy cannot know the message. Furthermore, Blaze *et al.* [3] gave two methods to classify PRE. One is according to the allowed times of transformation. If the ciphertext can be transformed from Alice to Bob, then from Bob to Charlie, and so on, then the PRE scheme is multi-use; otherwise, it is single use. The other method is according to the direction of transformation. If the re-encryption key can be used to transform the ciphertext from Alice to Bob, and vice versa, then the PRE scheme is bidirectional; otherwise, it is unidirectional.

Since the introduction of PRE by Blaze *et al.* [3], there have been many papers [1–11] that have proposed different PRE schemes with different security properties.

The first chosen-plaintext secure (CPA-secure), bidirectional, multi-use PRE scheme was proposed by Blaze *et al.* [3]; however, it is not collusion resistant.[†] Based on public key encryption with double trapdoors, Ateniese *et al.* [1,2] proposed the first collusion-resistant, unidirectional, single-use PRE schemes with CPA security. At the Theory of Cryptography Conference 2007, Hohenberger *et al.* [6] proposed a new collusion-resistant, CPA-secure, unidirectional PRE scheme, where the re-encryption key together with the re-encryption algorithm can be treated as an obfuscated re-encryption program.

However, many PRE's applications, such as the distributed file system, demand that the underlying PRE scheme is secure against chosen-ciphertext attack (CCA).

[†]Collusion resistance means that the delegatee colluding with the proxy cannot get the delegators secret key. See the definition in Section 2.

To solve the problem, Canetti and Hohenberger [7] and Green and Ateniese [5] proposed the definition of CCA security and some new CCA-secure PRE schemes independently. In the latter part of this paper, many other PRE schemes with some special properties are proposed, such as (R)CCA-secure[‡] scheme in the standard model [12–16], scheme without pairings [8–10,17], and scheme with invisible proxy [18,19].

Nevertheless, besides protecting the secrecy of the message in PRE, it is also useful to protect the secrecy of the recipients' identities [11]. For example, in a distributed file system [1,2], it might require that the proxy file server re-encrypts the sensitive files without knowing the recipients' identities. The server might acquire re-encryption keys from a re-encryption key pool but not directly from the delegators. And the server might be told to re-encrypt some files with some re-encryption key without being able to deduce the public keys behind these values. In this case, if the server is compromised, the adversary will not be able to extract a list of "Who was speaking privately with whom" [11] or a list of "Who is using the re-encryption service". This is highly desirable for some private communication scenarios. To the best of our knowledge, none of the mentioned PRE scheme has the anonymity property.

The anonymity property of PRE was first investigated by Ateniese *et al.* [11]. Their proposed anonymity includes two anonymity aspects: the anonymity of the re-encrypted ciphertext (i.e., the adversary cannot deduce the identity of the delegatee from the re-encrypted ciphertext, even if he/she has the re-encryption key) and the anonymity of the re-encryption key (i.e., the proxy (adversary) cannot decide whether the received re-encryption key is a real re-encryption key or a random value from the re-encryption key space). However, the given scenario additionally demands another anonymity aspect, the anonymity of the original ciphertext (i.e., the adversary cannot deduce the identity of the delegator from the original ciphertext, even if he/she has the re-encryption key).[§] If the original ciphertext is not anonymous, the adversary can easily extract a list of "Who is using the re-encryption service". Note that the anonymities of both the re-encryption key and original ciphertext imply the anonymity of the re-encrypted ciphertext. See the details in Remark 3. Although Ateniese *et al.* [11] proposed the first PRE scheme, which holds the anonymity property, they did not propose any CCA-secure PRE scheme with anonymity but left the problem for future work.

Motivated by the mentioned problems, we would like to study the anonymity of PRE in this paper. In particular, we extend the definition of anonymity proposed by Ateniese *et al.* [11]. This extension not only is from CPA-secure to CCA-secure scheme, but also has the following main differences compared with that in reference [11].

- Our definition covers all the three anonymity aspects, whereas the definition in reference [11] only covers the first two.
- Our definition allows that multiple re-encryption keys can be associated with one delegation, and that the adversary can query the re-encryption key associated with the challenge delegation when we deal with the first two anonymity aspects, whereas the definition in reference [11] does not allow these two situations.

Then, we propose the first PRE scheme satisfying our new definition. The proposal's security can be proven in the random oracle model under the DDH assumption. Unlike the scheme in reference [11], our proposal is without pairings.

In the rest of the paper, we first introduce some basic knowledge we used in this study. In the latter part of this paper, we further introduce the Schnorr signature and the modified ElGamal encryption, which are used in our proposal. In Section 4, we propose our construction and give the security proofs. In Section 5, we compare our proposal and the scheme in reference [11] in terms of computational cost, ciphertext size, and security level. Finally, we draw the conclusion.

2. DEFINITIONS AND SECURITY NOTIONS

In this section, we review and introduce some basic knowledge we used in this paper, including the definitions and security models for PRE.

2.1. Single-use unidirectional proxy re-encryption

Definition 1. *A single-use unidirectional PRE scheme (SUPRE) is a tuple of probably polynomial time (p.p.t.) algorithms (KeyGen, ReKeyGen, Enc, ReEnc, Dec)*:

- *KeyGen, Enc, and Dec*. Identical to those in the traditional public key encryption.
- *ReKeyGen* $(sk_1, pk_2) \rightarrow rk_{pk_1, pk_2}$.[¶] On inputting a secret key sk_1 and a public key pk_2, the re-encryption key generation algorithm ReKeyGen outputs a unidirectional re-encryption key rk_{pk_1, pk_2}.

[‡]Replayable CCA (RCCA) is a stronger attack than the chosen plaintext attack but a weaker attack than the chosen ciphertext attack.

[§]The scheme in reference [11] satisfies all these three anonymities under chosen-plaintext attacks, though their definition just covers the first two anonymities.

[¶]There are two kinds of `ReKeyGen` according to whether the delegatee is involved. If not, it is non-interactive; otherwise, it is interactive. In this paper, we only consider the non-interactive `ReKeyGen`.

- *ReEnc* $\left(rk_{pk_1,pk_2}, C_1\right) \rightarrow C_2$. On inputting a re-encryption key rk_{pk_1,pk_2} and a ciphertext C_1, the re-encryption algorithm ReEnc outputs a re-encrypted ciphertext C_2 or $\perp$.

Correctness. A correct PRE scheme should satisfy the following two requirements:

$$\begin{aligned} \mathrm{Dec}(sk_1, \mathrm{Enc}(pk_1, m)) &= m \\ \mathrm{Dec}(sk_2, \mathrm{ReEnc}(\mathrm{ReKeyGen}(sk_1, pk_2), C)) &= m \end{aligned}$$

where $(pk_1, sk_1), (pk_2, sk_2) \leftarrow \mathrm{KeyGen}(1^\lambda)$ and C is the ciphertext of message m under pk_1 from algorithm Enc.

2.2. Security notions for single-use unidirectional proxy re-encryption

2.2.1. Indistinguishability of encryptions under chosen-ciphertext attack for the original ciphertext of single-use unidirectional proxy re-encryption [5,7,9].

The Indistinguishability of Encryptions under CCA security for the Original ciphertext (IE-CCA-O) of SUPRE is defined by the following CCA game played between a challenger C and an adversary $\mathcal{A}$. Note that we work in the static corruption model, where the adversary must decide the corrupted users before the game starts. Furthermore, we assume that the public keys inputted into the oracles by the adversary are all from $\mathcal{O}_{pk}$.

Phase 1. The adversary $\mathcal{A}$ issues queries $q_1, \ldots, q_{n_1}$ adaptively where query q_i is one of the following:

- *Public key generation oracle* $\mathcal{O}_{pk}$. On inputting an index i,[‖] the challenger takes a security parameter λ and responds by running algorithm KeyGen(1^λ) to generate a key pair (pk_i, sk_i), gives pk_i to $\mathcal{A}$, and records (pk_i, sk_i) in the table T_K.
- *Secret key generation oracle* $\mathcal{O}_{sk}$. On inputting pk_i by $\mathcal{A}$, if pk_i is corrupted, the challenger searches pk_i in the table T_K and returns sk_i; otherwise, the challenger returns $\perp$.
- *Re-encryption key generation oracle* $\mathcal{O}_{rk}$. On inputting (pk_i, pk_j) by $\mathcal{A}$, the challenger returns the re-encryption key $rk_{pk_i,pk_j} = \mathrm{ReKeyGen}\left(sk_i, pk_j\right)$, where sk_i is the secret key corresponding to pk_i.
- *Re-encryption oracle* $\mathcal{O}_{re}$. On inputting (pk_i, pk_j, C) by $\mathcal{A}$, the re-encrypted ciphertext $C' = \mathrm{ReEnc}(\mathrm{ReKeyGen}(sk_i, pk_j), C)$ is returned by the challenger, where sk_i is the secret key corresponding to pk_j.
- *Decryption oracle* $\mathcal{O}_{dec}$. On inputting (pk_i, C_i), the challenger returns $\mathrm{Dec}(sk_i, C_i)$, where sk_i is the secret key corresponding to pk_i.

Challenge. Once $\mathcal{A}$ decides that phase 1 is over, it outputs two equal length plaintexts, m_0 and m_1, from the message space and a public key pk^* on which it wishes to challenge. There are two restrictions on the public key pk^*: (i) pk^* is an uncorrupted public key; (ii) if $(pk^*, \bigstar)$ did appear in any query to $\mathcal{O}_{rk}$, then $\bigstar$ should be uncorrupted. The challenger picks a random bit $\mathbf{b} \in \{0, 1\}$ and sets $C^* = \mathrm{Enc}(pk^*, m_{\mathbf{b}})$. It sends C^* as the challenge to $\mathcal{A}$.

Phase 2. The adversary $\mathcal{A}$ issues more queries $q_{n_1+1}, \cdots, q_n$ adaptively where query q_i is one of the following:

- $\mathcal{O}_{pk}$ *and* $\mathcal{O}_{sk}$. The challenger responds as in phase 1.
- $\mathcal{O}_{rk}$. On inputting (pk_i, pk_j) by $\mathcal{A}$, if $pk_i = pk^*$ and pk_j is corrupted, the challenger outputs $\perp$; otherwise, the challenger responds as in phase 1.
- $\mathcal{O}_{re}$. On inputting (pk_i, pk_j, C_i) by $\mathcal{A}$, if $(pk_i, C_i) = (pk^*, C^*)$ and pk_j is corrupted, the challenger outputs $\perp$; otherwise, the challenger responds as in phase 1.
- $\mathcal{O}_{dec}$. On inputting (pk_i, C_i), if (pk_i, C_i) is not a derivative[**] of (pk^*, C^*), the challenger outputs $\perp$; otherwise, the challenger responds as in phase 1.

Guess. Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}' \in \{0, 1\}$ and wins the game if $\mathbf{b} = \mathbf{b}'$.

The advantage $\mathbf{Adv}^{\mathrm{IE-CCA-O}}_{\mathrm{SUPRE}}(\lambda)$ is defined as $|\Pr[\mathbf{b} = \mathbf{b}'] - 1/2|$. The scheme PRE is said to be *IE-CCA-O* secure if for all efficient adversaries $\mathcal{A}$, the advantage $\mathbf{Adv}^{\mathrm{IE-CCA-O}}_{\mathrm{SUPRE}}(\lambda)$ is negligible.

2.2.2. Collusion resistance under chosen-ciphertext attack for single-use unidirectional proxy re-encryption [9].

The Collusion Resistance under CCA (CR-CCA) security for SUPRE is defined by the same method of the IE-CCA-O security for SUPRE.

Find. There exist public key generation oracle $\mathcal{O}_{pk}$, secret key generation oracle $\mathcal{O}_{sk}$, re-encryption key generation oracle $\mathcal{O}_{rk}$, and decryption oracle $\mathcal{O}_{dec}$ as that in phase 1 of the IE-CCA-O game. There is no re-encryption oracle in the CR-CCA game for SUPRE because the adversaries can obtain any re-encryption key they want, and they can use the obtained re-encryption key to generate the corresponding re-encrypted ciphertexts.

Output. Finally, the adversary $\mathcal{A}$ outputs a key sk and wins the game if sk is the secret key sk of an uncorrupted user.

The advantage $\mathbf{Adv}^{\mathrm{CR-CCA}}_{\mathrm{SUPRE}}(\lambda)$ is defined as $\Pr[\mathcal{A}$ wins$]$. The scheme PRE is said to be *CR-CCA* secure if for all

[‖]This index is just used to distinguish different public keys.

[**]Derivatives of (pk^*, C^*) are defined as follows. This definition is adapted from that in reference [7]:

(1) (pk^*, C^*) is a derivative of itself.
(2) If $\mathcal{A}$ has queried $\mathcal{O}_{re}$ on inputting (pk_i, pk_j, C_i) and obtained (pk_j, C_j), then (pk_j, C_j) is a derivative of (pk_i, C_i).
(3) If $\mathcal{A}$ has queried $\mathcal{O}_{rk}$ on inputting (pk_i, pk_j) and $C_j = \mathrm{ReEnc}\mathcal{O}_{\mathrm{rk}} pk_i, pk_j, C_i$, then (pk_j, C_j) is a derivative of (pk_i, C_i).

efficient adversaries $\mathcal{A}$, the advantage $\mathbf{Adv}_{\mathrm{SUPRE}}^{\mathrm{CR-CCA}}(\lambda)$ is negligible.

Remark 1: Indistinguishability of Encryptions under CCA for the Re-encrypted ciphertext (IE-CCA-R) security. In many previous papers [12,11], the CR-CCA security is replaced by another security, named IE-CCA-R security. The game of IE-CCA-R security is almost the same as that of IE-CCA-O security, except that there is no restriction on re-encryption key generation oracle, and that the challenge ciphertext is a re-encrypted ciphertext under the challenge public key. Clearly, the CR-CCA security is implied by the IE-CCA-R security, because the adversary, knowing the secret key, can definitely win the game of IE-CCA-R security.

2.2.3. Indistinguishability of keys under chosen-ciphertext attack for the original ciphertext of single-use unidirectional proxy re-encryption.

The Indistinguishability of Keys under CCA for the Original ciphertext (IK-CCA-O) security of SUPRE is defined by the same method of the IE-CCA-O security.

Phase 1. Identical to that in the IE-CCA-O game.

Challenge. Once the adversary $\mathcal{A}$ decides that phase 1 is over, it outputs two public keys, pk_0^* and pk_1^*, and one message m^*. The inputted two public keys should satisfy the following properties: (i) pk_0^* and pk_1^* are both uncorrupted. (ii) If $(pk_b^*, \bigstar)$ $(b \in \{0,1\})$ has been queried to the re-encryption key generation oracle, $\bigstar$ should be uncorrupted. The challenger picks a random bit $\mathbf{b} \in \mathbf{0}, 1$ and returns $C^* = \mathrm{Enc}(pk_{\mathbf{b}}^*, m^*)$ as the challenge ciphertext.

Phase 2. It runs almost the same as that in phase 1, but with the following restrictions:

- $\mathcal{O}_{rk}$. If the input (pk_i, pk_j) satisfies $pk_i = pk_b^*$ $(b \in \{0,1\})$, then pk_j should be uncorrupted. Note that the restriction is required, because if not, the adversary can win the game trivially. The adversary first acquires a derivative of (pk_b^*, C^*): (pk_j, C_j) by using the obtained re-encryption key and then decrypts C_j by using the secret key of pk_j. If the resulting message is m^*, then $\mathbf{b} = b$; otherwise, $\mathbf{b} = |b-1|$.
- $\mathcal{O}_{re}$. The input (pk_i, pk_j, C_i) cannot satisfy the following situations simultaneously: $pk_i = pk_b^*$ $(b \in \{0,1\})$, $C_i = C^*$, and pk_j is uncorrupted.

Note that the restriction is required, because if not, the adversary can win the game trivially. The adversary first acquires a derivative of (pk_b^*, C^*): (pk_j, C_j) from this oracle and then decrypts C_j by using the secret key of pk_j. If the resulting message is m^*, then $\mathbf{b} = b$; otherwise, $\mathbf{b} = |b-1|$.

- $\mathcal{O}_{dec}$. The input (pk_i, C_i) cannot be the derivative of the challenge ciphertext.

Guess. Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}' \in \{0,1\}$ and wins the game if $\mathbf{b} = \mathbf{b}'$.

The advantage $\mathbf{Adv}_{\mathrm{SUPRE}}^{\mathrm{IK-CCA-O}}(\lambda)$ is defined as $\Pr[\mathbf{b} = \mathbf{b}'] - 1/2$. The scheme PRE is said to be *IK-CCA-O* secure if for all efficient adversaries $\mathcal{A}$, the advantage $\mathbf{Adv}_{\mathrm{SUPRE}}^{\mathrm{IK-CCA-O}}(\lambda)$ is negligible.

2.2.4. Indistinguishability of keys under chosen-ciphertext attack for the re-encryption key of single-use unidirectional proxy re-encryption.

The Indistinguishability of Keys under CCA for the Re-encryption key (IK-CCA-R) security of SUPRE is defined by the same method of the IE-CCA-R security.

Phase 1. Identical to that in the *find* phase of the CR-CCA game.

Challenge. Once the adversary $\mathcal{A}$ decides that phase 1 is over, it outputs two public keys, pk_I and pk_J, on which it wishes to challenge, where pk_I and pk_J are two uncorrupted public keys. The challenger picks a random bit $\mathbf{b} \in \{0,1\}$. If $\mathbf{b} = 0$, then it sets rk^* as a random key from the re-encryption key space; otherwise, it sets $rk^* = \mathrm{ReKeyGen}(sk_I, pk_J)$, where sk_I is the corresponding secret key of pk_I. At last, the challenger sends rk^* as the challenge to $\mathcal{A}$.

Phase 2. It runs almost the same as that in phase 1, but with the following restrictions:

- $\mathcal{O}_{dec}$. The input (pk_i, C_i) cannot satisfy the following situations simultaneously:
  - $pk_i = pk_J$;
  - C_i is a re-encrypted ciphertext by the challenge re-encryption key.

Guess. Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}' \in \{0,1\}$ and wins the game if $\mathbf{b} = \mathbf{b}'$.

The advantage $\mathbf{Adv}_{\mathrm{SUPRE}}^{\mathrm{IK-CCA-R}}(\lambda)$ is defined as $\Pr[\mathbf{b} = \mathbf{b}'] - 1/2$. The scheme PRE is said to be *IK-CCA-R* secure if for all efficient adversaries $\mathcal{A}$, the advantage $\mathbf{Adv}_{\mathrm{SUPRE}}^{\mathrm{IK-CCA-R}}(\lambda)$ is negligible.

Remark 2: IK-CPA-R security. The mentioned game without decryption oracle corresponds to the anonymity game in reference [11] (Definiiton 2.5 in reference [11], IK-CPA-R security). In our game, the adversary is allowed to query the re-encryption key generation oracle with the challenge delegation (pk_I, pk_J), and multiple re-encryption keys are allowed in one delegation. However, these two situations are not allowed in reference [11]. As mentioned in reference [11] (Remark 3.5 in reference [11]), the scheme cannot be proven secure if these two situations are allowed.

Remark 3: Anonymity of the re-encrypted ciphertext. The anonymity of the re-encrypted ciphertext can be implied by the anonymity of the re-encryption key or original ciphertext. The implication from the anonymity of the re-encryption key to the anonymity of the re-encrypted ciphertext has already been discussed in reference [11]. Here, we show the implication from the anonymity of the original ciphertext to the anonymity of the re-encrypted ciphertext. If we do not have the anonymity of the re-encrypted ciphertext, the adversary can win the IK-CCA-O

game given as follows. The adversary obtains a re-encrypted ciphertext of C^* by using the re-encryption key from pk_0^* to pk, where pk is an uncorrupted public key. Because the re-encrypted ciphertext is not anonymous, the adversary can decide whether the re-encrypted ciphertext corresponds to pk. If so, then $\mathbf{b} = 1$; otherwise, $\mathbf{b} = 0$.

3. BASIC TOOLS

3.1. Schnorr signature

Schnorr signature is proposed by Schnorr [20], and it is existentially unforgeable. The system parameters of Schnorr signature are $(\mathbb{G}, g, q, H(\cdot))$, where $\mathbb{G}$ is a finite cyclic group with generator g of prime order q, and $H(\cdot)$ is a cryptographic hash function, $H(\cdot) : \{0,1\}^* \rightarrow Z_q^*$

- KeyGen$(1^\lambda) \rightarrow k$. The key generation algorithm outputs a signing key $sk \xleftarrow{R} Z_q^*$ and a verifying key $pk = g^{sk}$.
- Sign$(sk, m) \rightarrow \sigma$. On inputting a signing key sk and a message m, the signing algorithm outputs a signature $\sigma = (s, R)$, where $R = g^r$, $s = r + sk \cdot H(m \| R) \bmod q$, and r is a random number from Z_q^*.
- Verify$(pk, m, \sigma) \rightarrow m$. On inputting a verifying key pk, a message m, and a signature σ, the verifying algorithm outputs 1, if $g^s = R \cdot pk^{H(m \| R)}$; otherwise, it outputs 0.

3.2. The modified ElGamal encryption

The modified ElGamal encryption is a modification of ElGamal encryption [21] with Fujisaki-Okamoto transformation [22]. The system parameters of the modified ElGamal encryption are $(\mathbb{G}, g, q, H_1(\cdot), H_2(\cdot), \text{SKE})$, where $\mathbb{G}$ is a finite cyclic group with generator g of prime order q, $H_1(\cdot)$ and $H_2(\cdot)$ are cryptographic hash functions, $H_1(\cdot) : \{0,1\}^* \rightarrow Z_q^*$, $H_2(\cdot) : \{0,1\}^* \rightarrow \{0,1\}^\ell$, and ℓ is the bit length of CPA-secure symmetric key encryption SKE = (Enc, Dec).

- *KeyGen*$(1^\lambda) \rightarrow k$. The key generation algorithm outputs a secret key $sk \xleftarrow{R} Z_q^*$ and a public key $pk = g^{sk}$.
- Enc$(pk, m) \rightarrow \sigma$. On inputting a public key pk and a message m, the encryption algorithm outputs a ciphertext $C = (C_1, C_2)$, where $C_1 = g^{H_1(\sigma \| m)}$, $C_2 = \text{SKE.Enc}\left(H_2\left(pk^{H_1(\sigma \| m)}\right), \sigma \| m\right)$, and σ is a random number from Z_q^*.
- Dec$(sk, C) \rightarrow m$. On inputting a secret key sk, with a ciphertext C, the decryption algorithm computes $\sigma \| m = \text{SKE.Dec}\left(H_2\left(C_1^{sk}\right)\right)$. If $g^{H_1(\sigma \| m)} = C_1$, it outputs m; otherwise, it outputs $\perp$.

Theorem 1. *The modified ElGamal encryption is encryption distinguishable under CCA in the random oracle model based on the DDH assumption in $\mathbb{G}$ and the CPA security of SKE. In short, the modified ElGamal encryption is IE-CCA secure.*

This theorem can be proved by the same method as that in reference [22]. *We omit it here.*

Theorem 2. *The modified ElGamal encryption is key private under CCA in the random oracle model if the DDH problem is hard in $\mathbb{G}$, the underlying hash functions are collision resistant, and SKE is CPA secure. In short, the modified ElGamal encryption is IK-CCA secure.*

This theorem can be proved by the similar method with random oracles as the one proposed by Bellare et al. [23]. *We omit it here.*

4. OUR PROPOSAL

In this section, we first propose a new PRE scheme and then prove its CCA security, collusion resistance, and anonymity in the random oracle model one by one.

On the one hand, the method of obtaining CCA security of the original ciphertext is Schnorr signature [20]; that is, Schnorr signature guarantees the integrity of the original ciphertext.

On the other hand, to guarantee the CCA security of the re-encrypted ciphertext, several methods are used. As we will see in the next subsection, the re-encrypted ciphertext contains (A', B', D, U_1, U_2). The values of (A', B', D) are bound together by using α_2 and β_2 via the CCA security of SKE. α_2 and β_2 are computed from (U_1, U_2). (U_1, U_2) is a ciphertext under the delegatee's public key of the modified ElGamal encryption; hence, the adversary cannot obtain α_2 and β_2 because of the CCA security of the modified ElGamal encryption. As a result, (A', B', D, U_1, U_2) are bound together, and CCA security of the re-encrypted ciphertext is obtained.

Regarding the collusion resistance, we follow the method in reference [1,2], that is, public key encryption with double trapdoors. In particular, we have the strong secret key x and the weak secret key $(H_4(x \| 0) + x \cdot H_4(x \| 1))$, where $H_4(\cdot)$ is a hash function. The corresponding public keys are $pk_1 = g^x$ and $pk_2 = g^{H_4(x \| 0) + x \cdot H_4(x \| 1)}$, respectively. Furthermore, the values of $H_4(x \| 0)$ and $H_4(x \| 1)$ are shared between the delegatee and the proxy. This construction is original from the security proof in Cramer-Shoup scheme [24], where the simulator without knowing x but knowing $H_4(x \| 0)$ and $H_4(x \| 1)$ can answer the decryption queries correctly. This property helps us to answer the re-encryption oracle and decryption oracles in the security proof.

According to the IE-CCA-O game, only if the ciphertext output from algorithm Enc satisfies anonymity do we acquire the anonymity of the original ciphertext.

For the anonymity of the re-encryption key, we need to design a method to let the challenger be able to decide whether a re-encrypted ciphertext queried to $\mathcal{O}_{re}$ is computed by the challenge re-encryption key or not. Our solution is to let every PRE key include a unique value, which can be revealed during the decryption of the re-encrypted ciphertext. In particular, $rk^{(2)}$ in the re-encryption key[††] is such a value.

[††]See the details in algorithm `ReKeyGen` in Section 4.1.

Table I. The ideas in our proposal.

Goal	Method
Integrity of OC	FO transformation
Public verifiability of OC	Schnorr signature
Anonymity of OC	ElGamal encryption
Integrity of RC	CCA security of SKE
Anonymity of RK	Secret key splitting + ElGamal encryption
Collusion resistance	Cramer–Shoup encryption

CO, original ciphertext; RC, re-encrypted ciphertext; and RK, re-encrypted key.

We summarize the mentioned ideas in Table I.

4.1. The description of our proposal

Setup. The system parameters are $(g, q, \mathbb{G}, \text{SKE}, H_i(\cdot), i = 1, \cdots, 4)$, where $\mathbb{G}$ is a finite cyclic group with a prime order q, g is a generator of $\mathbb{G}$, SKE is a CCA-secure one-time symmetric key encryption scheme, and $H_i(\cdot)$ $(i = 1, \cdots, 4)$ are hash functions (treated as random oracles in the security proof),

$$H_1(\cdot) : \{0,1\}^* \to Z_q^*, \quad H_2(\cdot) : \mathbb{G} \to \{0,1\}^l,$$
$$H_3(\cdot) : \mathbb{G} \to Z_q^*, \quad H_4(\cdot) : \{0,1\}^* \to Z_q^*.$$

We denote l as the bit length of the secret key of SKE.

KeyGen. The user sets his public key as

$$\mathbf{pk} = (pk_1, pk_2) = \left(g^x, g^{H_4(x||0)+x\cdot H_4(x||1)}\right)$$

for random $x \xleftarrow{R} Z_q^*$.

ReKeyGen. On inputting a public key $\mathbf{pk}' = \left(pk_1', pk_2'\right) = \left(g^{x'}, g^{H_4(x'||0)+x'\cdot H_4(x'||1)}\right)$ and a secret key $\mathbf{sk} = x$, it outputs the unidirectional re-encryption key

$$rk_{\mathbf{pk},\mathbf{pk}'} = \left(rk^{(1)}_{\mathbf{pk},\mathbf{pk}'}, rk^{(2)}_{\mathbf{pk},\mathbf{pk}'}\right)$$

which is computed as follows:

(1) Choose random numbers $\alpha_1, \alpha_2, \beta_1, \beta_2 \in Z_q^*$, such that $\alpha_1 \cdot \alpha_2 = H_4(x\|0) \bmod q$ and $\beta_1 \cdot \beta_2 = H_4(x\|1) \bmod q$.
(2) Choose random number $\sigma \in Z_q^*$ and compute $k_{rk} = H_1(\sigma\|\alpha_2\|\beta_2)$.
(3) Set $rk^{(1)}_{\mathbf{pk},\mathbf{pk}'} = (\alpha_1, \beta_1)$.
(4) Compute

$$rk^{(2)}_{\mathbf{pk},\mathbf{pk}'} = (U_1, U_2) = \left(g^{k_{rk}}, \text{SKE.Enc}\left(H_2\left(\left(pk_1'\right)^{k_{rk}}\right), \sigma|\alpha_2|\beta_2\right)\right)$$

Note that (U_1, U_2) is computed as that in the encryption algorithm of the modified ElGamal encryption scheme. Here, the public key is $\left(g, pk_1'\right)$ and the message is $\alpha_2\|\beta_2$.

Enc. On inputting a public key

$$\mathbf{pk} = (pk_1, pk_2) = \left(g^x, g^{H_4(x||0)+x\cdot H_4(x||1)}\right)$$

and a message $m \in \{0,1\}^n$, the encryptor does the following operations:

(1) Choose random numbers $r, r' \xleftarrow{R} Z_q^*$.
(2) Compute

$$A = g^r, \quad B = pk_1^r, \quad C = g^{r'}$$
$$k = H_2\left(pk_2^r\right), \quad D = \text{SKE.Enc}(k, m)$$
$$S = H_3(A\|B\|C\|D) \cdot r + r' \bmod q$$

(3) Output the ciphertext $K = (A, B, C, D, S)$.

Note that (C, S) is computed as that in the signing algorithm of Schnorr signature. Here, the signing key is r and the message is (A, B, D).

ReEnc. On inputting a re-encryption key

$$rk_{\mathbf{pk},\ \mathbf{pk}'} = \left(rk^{(1)}_{\mathbf{pk},\ \mathbf{pk}'}, rk^{(2)}_{\mathbf{pk},\ \mathbf{pk}'}\right)$$

and a ciphertext $K = (A, B, C, D, S)$ under public key $\mathbf{pk}$. The proxy does the following operations:

- If $g^S = A^{H_3(A||B||C||D)} \cdot C$ holds, go to the next steps; otherwise, output $\perp$ and halt.
- Compute $A' = A^{\alpha_1}, \quad B' = B^{\beta_1}$.
- Output the re-encrypted ciphertext

$$K' = (A', B', D, U_1, U_2)$$

Dec. On inputting a secret key $\mathbf{sk}$ and any ciphertext K

- If K is an original ciphertext, such that $K = (A, B, C, D, S)$, the decryptor does the following operations. Note that in this case, we denote $\mathbf{sk} = \mathbf{x}$.
  - If $g^S = A^{H_3(A||B||C||D)} \cdot C$ holds, go to the next step; otherwise, output $\perp$ and halt.
  - Output

$$m = \text{SKE.Dec}\left(H_2\left(A^{H_4(x||0)} B^{H_4(x||1)}\right), D\right)$$

  - Note that m may be $\perp$.
- If K is a re-encrypted ciphertext, such that $K = (A', B', D, U_1, U_2)$, the decryptor does the following operations. Note that in this case, we denote $\mathbf{sk} = x'$.
  - Compute

$$\sigma|\alpha_2|\beta_2 = \text{SKE.Dec}\left(H_2\left(U_1^{x'}\right), U_2\right)$$

  - If $g^{H_1(\sigma||\alpha_2||\beta_2)} = U_1$ holds, output

$$m = \mathrm{SKE.Dec}\left(H_2\left(A'^{\alpha_2} \cdot B'^{\beta_2}\right), D\right)$$

- Note that m may be $\perp$; otherwise, output $\perp$.

Correctness. Because of the following equations, we have the correctness of our proposal.

- If $K = (A, B, C, D, S)$, we have

$$A^{H_4(x||0)} \cdot B^{H_4(x||1)} = g^{r(H_4(x||0)+x \cdot H_4(x||1))} = pk_2^r$$

- If $K = (A', B', D, U_1, U_2)$, we have

$$U_1^{x'} = g^{k_{rk} \cdot x'} = \left(pk_1'\right)^{k_{rk}}$$

- and

$$\begin{aligned} A'^{\alpha_2} B'^{\beta_2} &= \left(A^{\alpha_1}\right)^{\alpha_2} \cdot \left(B^{\beta_1}\right)^{\beta_2} \\ &= A^{\alpha_1 \cdot \alpha_2} \cdot B^{\beta_1 \cdot \beta_2} \\ &= g^{r \cdot H_4(x||0)} \cdot g^{r \cdot x \cdot H_4(x||1)} \\ &= pk_2^r \end{aligned}$$

4.2. Security analysis of our proposal

Theorem 3. *Our proposal is IE-CCA-O secure in the random oracle model under assumptions that the DDH problem is hard, that SKE is CCA secure, and that Schnorr signature is existentially unforgeable. In particular, we have that*

$$\mathbf{Adv}_{\mathrm{SUPRE}}^{\mathrm{IK-CCA-}O}(\lambda) \leq \frac{q_{H_1} + q_{H_3} + q_{rk}}{q} + \frac{q_{H_2} + q_{rk}}{2^{\updownarrow}} + \varepsilon_{\mathrm{DDH}} + \varepsilon_{\mathrm{Sch}} + \varepsilon_{\mathrm{SKE}}$$

where q_{H_1} is the maximum number of queries to H_1 oracle, q_{H_2} is the maximum number of queries to H_2 oracle, q_{H_3} is the maximum number of queries to H_3 oracle, *q_{rk} is the maximum number of queries to the re-encryption key generation oracle, $\varepsilon_{\mathrm{DDH}}$ is the probability that the adversary solves the DDH problem, $\varepsilon_{\mathrm{Sch}}$ is the probability that the adversary breaks the existential unforgeability of Schnorr signature, and $\varepsilon_{\mathrm{SKE}}$ is the probability that the adversary breaks the CCA security of the underlying SKE.*

Proof. Assume that there is an adversary $\mathcal{A}$ that can break the IE-CCA-O security of our proposal, then we can build an algorithm $\mathcal{B}$ that solves the DDH problem by using $\mathcal{A}$. On inputting (g, g^a, g^b, T), $\mathcal{B}$ decides whether $T = g^{ab}$.

Phase 1. We have the following oracles.

Random oracles:

- $\mathcal{O}_{H_1}$. On inputting $(\sigma, \alpha_2, \beta_2) \in Z_q^{*3}$, it first checks whether there exists a tuple $(\sigma, \alpha_2, \beta_2, r^{(1)})$ in the table T_{H_1}. If it exists, return $r^{(1)}$. If it does not exist, $\mathcal{B}$ chooses a random number $r^{(1)}$ from Z_q^*, returns $r^{(1)}$, and records $(\sigma, \alpha_2, \beta_2, r^{(1)})$ in the table T_{H_1}.
- $\mathcal{O}_{H_2}$. On inputting $R \in \mathbb{G}$, it first checks whether there exists a pair $(R, r^{(2)})$ in the table T_{H_2}. If it exists, return $r^{(2)}$. If it does not exist, $\mathcal{B}$ chooses a random number $r^{(2)}$ from $\{0,1\}^{\ell}$, returns $r^{(2)}$, and records $(R, r^{(2)})$ in the table T_{H_2}.
- $\mathcal{O}_{H_3}$. On inputting $(A, B, C, D) \in \mathbb{G}^3 \times C$, where C is the ciphertext space of SKE, it first checks whether there exists a tuple $(A, B, C, D, r^{(3)})$ in the table T_{H_3}. If it exists, return $r^{(3)}$. If it does not exist, $\mathcal{B}$ chooses a random number $r^{(3)}$ from Z_q^*, returns $r^{(3)}$, and records $(A, B, C, D, r^{(3)})$ in the table T_{H_3}.
- $\mathcal{O}_{H_4}$. On inputting (R, κ), where $R \in Z_q^*$ and $\kappa \in \{0,1\}$, $\mathcal{B}$ first checks whether there exists a tuple $(R, \kappa, r^{(4)})$ in the table T_{H_4}.
  - If it exists, return $r^{(4)}$.
  - If it does not exist, $\mathcal{B}$ checks whether $g^{R \cdot x_{i1}} = pk_{i1}$ holds, where x_{i1}, pk_{i1} are the values in the table T_{uk} (see $\mathcal{O}_{pk}$). If it holds, then $\mathcal{B}$ acquires $a = R$ and is used to solve the DDH problem.
  - Otherwise, $\mathcal{B}$ chooses a random number $r^{(5)}$ from Z_q^*, returns $r^{(4)}$, and records $(R, \kappa, r^{(4)})$ in the table T_{H_4}.

Public key generation oracle $\mathcal{O}_{\mathrm{pk}}$. We have two cases in this oracle:

- If it is a corrupted public key, set $\mathbf{pk}_i = (pk_{i1}, pk_{i2}) = \left(g^{x_i}, g^{H_4(x_i||0)+x_i \cdot H_4(x_i||1)}\right)$ and record (pk_{i1}, pk_{i2}, x_i) into the table T_{ck}, where x_i is a random number from Z_q^*.
- If it is an uncorrupted public key, set $\mathbf{pk}_i = (pk_{i1}, pk_{i2}) = \left((g^a)^{x_{i1}},\ g^{x_{i2}} \cdot (g^a)^{x_{i1} \cdot x_{i3}}\right)$ and record $(pk_{i1}, pk_{i2}, x_{i1}, x_{i2}, x_{i3})$ into the table T_{uk}, where x_{i1}, x_{i2}, and x_{i3} are random numbers from Z_q^*.

Secret key generation oracle $\mathcal{O}_{sk}$. On inputting $\mathbf{pk}_i = (pk_{i1}, pk_{i2})$ by the adversary, if it is a corrupted public key, $\mathcal{B}$ searches tuple (pk_{i1}, pk_{i2}, x_i) in the table T_{ck} and returns the corresponding x_i to the adversary; otherwise, $\mathcal{B}$ returns $\perp$.

Re-encryption key generation oracle $\mathcal{O}_{rk}$. On inputting $(\mathbf{pk}_i, \mathbf{pk}_j)$, where $\mathbf{pk}_i = (pk_{i1}, pk_{i2})$, $\mathbf{pk}_j = (pk_{j1}, pk_{j2})$, and $\mathbf{pk}_i$ and $\mathbf{pk}_j$ are both from $\mathcal{O}_{pk}$:

- If $\mathbf{pk}_i$ is corrupted, $\mathcal{B}$ first searches the tuple corresponding to $\mathbf{pk}_i$ in the table T_{ck} and acquires the values of x_i to do the same operations as in the real execution.
- If $\mathbf{pk}_i$ is uncorrupted, $\mathcal{B}$ first searches the tuple corresponding to $\mathbf{pk}_i$ in the table T_{uk}, acquires the values of (x_{i1}, x_{i2}, x_{i3}) to set $\alpha_1 \cdot \alpha_2 = x_{i2}$ mod q and $\beta_1 \cdot \beta_2 = x_{i3}$ mod q, and then does steps 2–4 in ReKeyGen to acquire the re-encryption key. At last, $\mathcal{B}$ returns the resulting re-encryption key.

Re-encryption oracle $\mathcal{O}_{re}$. On inputting $(\mathbf{pk}_i, \mathbf{pk}_j, K_i)$, $\mathcal{B}$ returns $\mathrm{ReEnc}\left(\mathcal{O}_{\mathrm{rk}}\left(\mathbf{pk}_i, \mathbf{pk}_j\right), K_i\right)$.

Decryption oracle $\mathcal{O}_{dec}$. On inputting $(\mathbf{pk}_i, K_i)$,

- If $\mathbf{pk}_i$ is corrupted, $\mathcal{B}$ acquires the corresponding x_i from the table T_{ck} and does the same operations as in the real execution.
- If $\mathbf{pk}_i$ is uncorrupted and $K_i = (A, B, C, D, S)$, $\mathcal{B}$ returns $\mathrm{Dec}(\mathbf{sk}', \mathrm{ReEnc}(\mathcal{O}_{\mathrm{rk}}(\mathbf{pk}_i, \mathbf{pk}'),\ \mathrm{K}))$, where $\mathbf{pk}'$ is corrupted and $\mathbf{sk}'$ is the associated secret key.

- If $\mathbf{pk}_i$ is uncorrupted and $K_i=(A', B', D, U_1, U_2)$, $\mathcal{B}$ does the following operations.

  - Find the tuple $(\sigma, \alpha_2, \beta_2, r^{(1)})$ in the table T_{H_1} and the tuple $(R, r^{(2)})$ in the table T_{H_2}, such that $\sigma\|\alpha_2\|\beta_2 =$ SKE. Dec$(r^{(2)}, U_2)$, $g^{r^{(1)}} = U_1$. If such tuples exist, do the next step; otherwise, output $\perp$ and halt.
  - Output $m =$ SKE.Dec$\left(H_2\left(A'^{\alpha_2}\cdot B'^{\beta_2}\right),\ D\right)$. Note that m may be $\perp$.

Challenge. Once the adversary $\mathcal{A}$ decides that phase 1 is over, it outputs plaintexts m_0, m_1, and a public key pk* with the restrictions specified in the IE-CCA-O game. $\mathcal{B}$ acquires $\left(x_1^*, x_2^*, x_3^*\right)$ corresponding to pk* from the table T_{uk}, chooses a random $\mathbf{b}\in\{0,1\}$, and sets

$$A^* = g^b, \quad B^* = T^{x_1^*}$$

$$k^* = H_2\left(\left(g^b\right)^{x_2^*}\cdot T^{x_1^* x_3^*}\right),\ \ C^* = \text{SKE.Enc}(k^*, m_{\mathbf{b}})$$

Then $\mathcal{B}$ chooses random numbers $S^*, r^{(3)*}$, computes $B^* = g^{S^*}/\left(g^b\right)^{r^{(3)}*}$, and records $\left(A^*, B^*, C^*, D^*, r^{(3)*}\right)$ into the table T_{H_3}. If $(A^*, B^*, C^*, D^*, \bigstar)$ has already existed in the table T_{H_3}, $\mathcal{B}$ reports "failure" and halts; otherwise, $\mathcal{B}$ returns $(A^*, B^*, C^*, D^*, S^*)$ as the challenge ciphertext.

Note that if $T=g^{ab}$, then we have $B^* = \left(g^{ab}\right)^{x_1^*} = \left(pk_1^*\right)^b$ and $\left(g^b\right)^{x_2^*}\cdot T^{x_1^* x_3^*} = \left(g^b\right)^{x_2^*}\cdot\left(g^{ab}\right)^{x_1^* x_3^*} = \left(pk_2^*\right)^b$.

Phase 2. The adversary $\mathcal{A}$ issues more queries as it does in phase 1 but with the restrictions specified in the IE-CCA-O game. $\mathcal{B}$ responds the queries as it does in phase 1.

Guess. Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}'\in\{0,1\}$. If $\mathbf{b}=\mathbf{b}'$, then (g, g^a, g^b, T) is a DDH tuple; otherwise, it is not a DDH tuple.

First, we analyze the probability that the simulation is indistinguishable from the real execution.

- It is clear that random oracles, public key oracle, secret key oracle, re-encryption key generation oracle, and re-encryption oracle in phase 1 and phase 2 are perfect.
- The decryption oracles in phase 1 and phase 2 are perfect except that the adversary does not query hash oracles (H_1, H_2) but guessed the right values. These issues happen with probabilities $\frac{q_{H_1}+q_{rk}}{q}$ and $\frac{q_{H_2}+q_{rk}}{2^t}$ at most, respectively.
- Only when $\mathcal{B}$ reports "failure" is the challenge phase of the simulation distinguishable from the real execution. This issue happens with the probability q_{H_3}/q at most.

Second,

- If the adversary can break the existential unforgeability of Schnorr signature, then it can generate a ciphertext such as (A^*, B^*, C, D^*, S), where $C\neq C^*$ and $S\neq S^*$. In this case, the adversary simply queries decryption oracle with (A^*, B^*, C, D^*, S) to acquire $m_{\mathbf{b}}$.
- If the adversary can break the CCA security of SKE, then it can acquire $\mathbf{b}$ from the challenge ciphertext with this ability.

Hence, we acquire the result as required. □

Theorem 4. *Our proposal is CR-CCA secure in the random oracle model under the discrete logarithm (DL) assumption in* $\mathbb{G}$. *In particular, we have that*

$$\mathbf{Adv}_{\text{SUPRE}}^{\text{CR-CCA}}(\lambda)\leq\frac{q_{H_1}+q_{rk}}{q}+\frac{q_{H_2}+q_{rk}}{2^t}+\varepsilon_{\text{DL}}$$

where q_{H_1}, q_{H_2}, *and* q_{rk} *are the same as that in Theorem 3 and* ε_{DL} *is the probability that the adversary solves the DL problem in* $\mathbb{G}$.

Proof. Assume that there is an adversary $\mathcal{A}$ that can break the CR-CCA security of our proposal, then we can build an algorithm $\mathcal{B}$ that solves the DL problem by using A. On inputting (g, g^a), $\mathcal{B}$ aims to output a.

Find. There exist the same random oracles, $\mathcal{O}_{pk}$, $\mathcal{O}_{sk}$, $\mathcal{O}_{rk}$, and $\mathcal{O}_{dec}$ as that in phase 1 in the proof of Theorem 3.

Output. The adversary outputs sk $= x$ of an uncorrupted public key pk_i and $\mathcal{B}$ outputs x/x_{i1} as the solution of the DL problem, where x_{i1} is the value corresponding to pk_i in the table T_{uk}.

As analyzed in the proof of Theorem 3, except the decryption oracle, other oracles are perfect. Hence, we obtain the result as required. □

Remark 4. Our proposal can be proven IE-CCA-R secure in the random oracle under the DDH assumption by using the similar method in the proof of Theorem 3. We omit it here.

Theorem 5. *Our proposal is IK-CCA-O secure in the random oracle model under the assumptions that the DDH problem is hard, that SKE is CCA secure, and that Schnorr signature is existentially unforgeable. In particular, we have that*

$$\mathbf{Adv}_{\text{SUPRE}}^{\text{IK-CCA-O}}(\lambda)\leq\frac{q_{H_1}+q_{H_3}+q_{rk}}{q}+\frac{q_{H_2}+q_{rk}}{2^t}+\varepsilon_{\text{DDH}}+\varepsilon_{\text{Sch}}+\varepsilon_{\text{SKE}}$$

where the notations are the same as that in Theorem 3.

Proof. Assume that there is an adversary $\mathcal{A}$ that can break the IK-CCA-O security of our proposal, then we can build an algorithm $\mathcal{B}$ that solves the DDH problem by using $\mathcal{A}$. On inputting (g, g^a, g^b, T), $\mathcal{B}$ decides whether $T=g^{ab}$.

Phase 1. The oracles in this phase are identical to those in the proof of Theorem 3.

Challenge. Once the adversary $\mathcal{A}$ decides that phase 1 is over, it outputs uncorrupted public keys pk_0^*, pk_1^*, and a message m^*. If this input does not satisfy the requirements specified in the security model, $\mathcal{B}$ aborts; otherwise, $\mathcal{B}$ acquires $\left(x_{i1}^*, x_{i2}^*, x_{i3}^*\right)$ corresponding to $\mathbf{pk}_i^*$ $(i=0,1)$ from the table T_{uk}, chooses a random $\mathbf{b}\in\{0,1\}$, and sets

$$A^* = g^b, , B^* = T^{x_{\mathbf{b}1}^*}$$

$$k^* = H_2\left(\left(g^b\right)^{x_{\mathbf{b}2}^*}\cdot T^{x_{\mathbf{b}1}^* x_{\mathbf{b}3}^*}\right)$$

$$C^* = \text{SKE. Enc}(k^*, m^*)$$

Then $\mathcal{B}$ chooses random numbers S^*, $r^{(3)^*}$, computes $B^* = g^{S^*}/\left(g^b\right)^{r^{(3)^*}}$, and records $\left(A^*, B^*, C^*, D^*, r^{(3)^*}\right)$ into the table T_{H_3}. If $(A^*, B^*, C^*, D^*, \bigstar)$ has already existed in the table T_{H_3}, $\mathcal{B}$ reports "failure" and halts; otherwise, $\mathcal{B}$ returns $(A^*, B^*, C^*, D^*, S^*)$ as the challenge ciphertext.

Note that if $T = g^{ab}$, then we have $B^* = \left(g^{ab}\right)^{x^*_{\mathbf{b}1}} = \left(pk^*_{\mathbf{b}1}\right)^b$ and $\left(g^b\right)^{x^*_{\mathbf{b}2}} \cdot T^{x^*_{\mathbf{b}1} x^*_3} = \left(g^b\right)^{x^*_{\mathbf{b}2}} \cdot \left(g^{ab}\right)^{x^*_{\mathbf{b}1} x^*_{\mathbf{b}3}} = \left(pk_{\mathbf{b}} 2^*\right)^b$.

Phase 2. The adversary $\mathcal{A}$ issues more queries as it does in phase 1 but with the restrictions specified in the IK-CCA-O game. $\mathcal{B}$ responds the queries as it does in phase 1.

Guess. Finally, the adversary $\mathcal{A}$ outputs a guess $\mathbf{b}' \in \{0, 1\}$. If $\mathrm{b} = \mathbf{b}'$, then (g, g^a, g^b, T) is a DDH tuple; otherwise, it is not a DDH tuple.

As analyzed in the proof of Theorem 3, we obtain the result as required. □

Theorem 6. *Our proposal is IK-CCA-R secure in the random oracle model under the assumptions that the modified ElGamal encryption scheme is IK-CCA secure in the random oracle model and that the DDH problem is hard in* $\mathbb{G}$. *In particular, we have that*

$$\mathbf{Adv}^{\mathrm{IK-CCA-R}}_{\mathrm{SUPRE}}(\lambda) \leq \frac{q_{H_1} + q_{rk}}{q} + \frac{q_{H_2} + q_{rk}}{2^l} + \varepsilon_{\mathrm{DDH}} + \varepsilon^{\mathrm{IK-CCA}}_{\mathrm{ElG}}$$

where the notations are the same as that in Theorem 3.

Proof. We incrementally define a sequence of games that start at the real attack (Game G_0) and end up at game G_3, which clearly shows that the adversary cannot break the system. We define E_i to be the event that $\mathbf{b} = \mathbf{b}'$ in game G_i, where $\mathbf{b}$ is the bit involved in the challenge phase and $\mathbf{b}'$ is the output of $\mathcal{A}$ in the guess phase.

Game G_0. This game corresponds to the real attack (the IK-CCA-R game). By definition,

$$|\Pr[E_0] - 1/2| = \mathbf{Adv}^{\mathrm{IK-CCA-R}}_{\mathrm{SUPRE}}(\lambda) \tag{1}$$

Game G_1. In this game, we replace the hash functions with the random oracles in the proof of Theorem 3. It is clear that

$$Pr[E_0] = Pr[E_1] \tag{2}$$

Game G_2. In this game, we replace public key oracle, secret key oracle, re-encryption key generation oracle, and decryption oracle with the ones in the phase 1 in the proof of Theorem 3. As analyzed in the proof of Theorem 3, we have that

$$|\Pr[E_2] - \Pr[E_1]| \leq \frac{q_{H_1} + q_{rk}}{q} + \frac{q_{H_2} + q_{rk}}{2^t} \tag{3}$$

Game G_3. In this game, we modify the challenge phase as follows.

Challenge. On inputting $(\mathbf{pk}_I, \mathbf{pk}_J) = ((pk_{I1}, pk_{I2}), (pk_{J1}, pk_{J2}))$ by the adversary, the challenger chooses a random bit $\mathbf{b}$. If $\mathbf{b} = 1$, the challenger first searches the tuple corresponding to $\mathbf{pk}_I$ in the table T_{uk} and acquires the values of (x_{i1}, x_{i2}, x_{i3}) to perform steps 2 and 3 in ReKeyGen. For step 4, we use pk_{I1} instead of pk_{J1}. We denote the result as $\left(U^*_1, U^*_2\right)$. The other operations are the same as that in game G_2.

According to the restriction in the decryption oracle in the IK-CCA-R game, the challenger does not need to decrypt $\left(U^*_1, U^*_2\right)$ to acquire $\sigma \| \alpha_2 \| \beta_2$. Hence, it is easy to see that if the adversary can distinguish game G_3 from game G_2, then we can build an algorithm that breaks IK-CCA security of the modified ElGamal encryption scheme by using the adversary. Here, the challenge message m^* for the modified ElGamal encryption scheme is $\sigma \| \alpha_2 \| \beta_2$, and the two challenge public keys are pk_{I1} and pk_{J1}. As a result, we have that

$$|\Pr[E_3] - \Pr[E_2]| \leq \varepsilon^{\mathrm{IK-CCA}}_{\mathrm{ElG}} \tag{4}$$

Furthermore, in game G_3, it is easy to see that when $\mathbf{b} = 1$, the challenge re-encryption key is a random value from the re-encryption key space. Hence, we have that

$$\Pr[E_3] = 1/2 \tag{5}$$

Combining Equations (1),(2),(3),(4),(5) we obtain the result as required. □

5. COMPARISON

In this section, we compare our proposal with scheme ABH09 [11] in terms of computational cost, ciphertext size, and security level (Table II).

Compared with exponentiations and pairings in finite cyclic groups, time spent on multiplications in finite cyclic groups, hash function, and symmetric key encryption is negligible; hence, we only consider time spent on exponentiations and pairings in Table II. The benchmarks used in Table II are from references [25,26], and time spent on one exponentiation in $\mathbb{G}$, one exponentiation in $\mathbb{G}_T$, one pairing, and one exponentiation in $\mathbb{G}$ are 6.4, 0.6, 5.9, and 1.8 ms, respectively. These benchmarks are obtained on workstation with a 64-bit, 3.2-GHz Pentium 4 processor. Furthermore, groups $\mathbb{G}, \mathbb{G}_T, G$ all have a 160-bit order, and one element in $\mathbb{G}, \mathbb{G}_T$ needs 512-bit storage and one element in $\mathbb{G}$ needs 1024-bit storage. We assume that one message needs 512-bit storage.

From Table II, we can see that scheme ABH09 needs less storage for ciphertexts and less decryption time for the re-encrypted ciphertext than our proposal, whereas our proposal are more efficient in algorithms ReKeyGen, Enc, ReEnc, Dec (for the original ciphertext). Our proposal cannot be proven secure in the stand model as scheme ABH09, but our proposal can satisfy a stronger security than scheme ABH09 (CCA security versus CPA security).

Table II. Comparison between our proposal and scheme ABH09.

Schemes			ABH09	Ours
Computational cost	ReKeyGen		19.9 ms	3.6 ms
	Enc		13.4 ms	7.2 ms
	ReEnc		24.8 ms	7.2 ms
	Dec	Original ciphertext	12.4 ms	7.2 ms
		Re-encrypted ciphertext	6.4 ms	7.2 ms
Ciphertext size	Original ciphertext		1536 bit	3904 bit
	Re-encrypted ciphertext		1024 bit	4384 bit
Security	Security level		IE-CPA, IK-CPA[a]	IE-CCA, IK-CCA
	Standard model		Yes	No
	Underlying assumptions		Extended Decisional Bilinear Diffie-Hellman, decision linear	DDH

[a] Scheme ABH09 can be proven secure in the model in reference [11] but not the one in Section 2.

6. CONCLUSION

In this paper, we revised the definition of anonymity for SUPRE. The revised definition guarantees anonymities of the re-encrypted ciphertext, re-encryption key and original ciphertext under the CCA. After that, based on the DDH assumption, we proposed the *first* PRE scheme, which is CCA secure, collusion resistant, and anonymous in the *random oracle model*.

In our proposal, the size of ciphertext will be extended after re-encryption, and the re-encrypted ciphertext needs more time to decrypt than the original ciphertext does. Hence, it is desired to design a CCA-secure, collusion-resistant, and anonymous PRE scheme without ciphertext's extension or decryption time's extension. To the best of our knowledge, homomorphic encryption [27] would be a good candidate. Because the ciphertext in homomorphic encryption can be transformed multi-time without any ciphertext's extension or decryption time's extension, we leave it for future research.

ACKNOWLEDGEMENT

The authors thank the anonymous reviewers for their insightful comments and helpful suggestions. Jun Shao was supported by NSFC No. 61003308, ZJGSUSF No. 1130XJ2010045, and ECZJF No. Y201017312. Peng Liu was supported by NSF CNS-0905131 and NSF CNS-0916469. Guiyi Wei was supported by NSFC No. 60803161.

REFERENCES

1. Ateniese G, Fu K, Green M, Hohenberger S. Improved proxy re-encryption schemes with applications to secure distributed storage. In *NDSS 2005, San Diego, California, USA. The Internet Society 2005*, 2005; 29–43.
2. Ateniese G, Fu K, Green M, Hohenberger S. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security (TISSEC)* 2006; **9**(1): 1–30.
3. Blaze M, Bleumer G, Strauss M. Divertible protocols and atomic proxy cryptography. In *EUROCRYPT 1998, LNCS*, Vol. **1403**. Springer: Heidelberg, 1998; 127–144.
4. Ivan A, Dodis Y. Proxy cryptography revisited. In *NDSS 2003, San Diego, California, USA. The Internet Society 2003*, 2003; 174–193.
5. Green M, Ateniese G. Identity-based proxy re-encryption. In *ACNS 2007, LNCS*, Vol. **4521**. Springer: Heidelberg, 2007; 288–306. Full version: Cryptology ePrint Archieve: Report 2006/473.
6. Hohenberger S, Rothblum G, Shelat A, Vaikuntanathan V. Securely obfuscating re-encryption. In *TCC 2007, LNCS*, Vol. **4392**. Springer: Heidelberg, 2007; 233–252.
7. Canetti R, Hohenberger S. Chosen-ciphertext secure proxy re-encryption. In *ACM CCS 2007*, 2007; 185–194. ACM New York, NY, USA. Full version: Cryptology ePrint Archieve: Report 2007/171.
8. Weng J, Deng RH, Liu S, Chen K, Lai J, Wang X. Chosen-ciphertext secure proxy re-encryption schemes without pairings. In *CANS 2008, LNCS*, Vol. **5339**. Springer: Heidelberg, 2008; 1–17.
9. Shao J, Cao Z. CCA-secure proxy re-encryption without pairings. In *PKC 2009, LNCS*, Vol. **5443**. Springer: Heidelberg, 2009; 357–376.
10. Chow S, Weng J, Yang Y, Deng R. Efficient unidirectional proxy re-encryption. In *Africacrypt 2010, LNCS*, Vol. **6055**. Springer: Heidelberg, 2010; 316–332. Full version: http://eprint.iacr.org/2009/189
11. Ateniese G, Benson K, Hohenberger S. Key-private proxy re-encryption. In *RSA 2009, LNCS*, Vol. **5473**. Springer: Heidelberg, 2009; 279–294.

12. Libert B, Vergnaud D. Unidirectional chosen-ciphertext secure proxy re-encryption. In *PKC 2008, LNCS*, Vol. **4939**. Springer: Heidelberg, 2008; 360–379.
13. Weng J, Chen M, Yang Y, Deng R, Chen K, Bao F. CCA-secure unidirectional proxy re-encryption in the adaptive corruption model without random oracles. *Science China Information Sciences* 2010; **53**(3): 593– 606. Updated version: http://eprint.iacr.org/2010/265
14. Shao J, Cao Z, Liu P. SCCR: a generic approach to simultaneously achieve CCA security and collusion-resistance in proxy re-encryption. *Security and Communication Networks* 2011; **4**(2): 122–135.
15. Shao J, Cao Z, Liu P. CCA-secure PRE scheme without random oracles; Cryptology ePrint Archive http://eprint.iacr.org/2010/112
16. Zhao G, Fang L, Wang J, Ge C, Ren Y. Improved unidirectional chosen-ciphertext secure proxy re-encryption. In *2010 IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, Beijing, China 2010; 476–480.
17. Zhang L, Wang XA, Wang W. Toward CCA-secure bidirectional proxy re-encryption without pairing nor random oracle. In *2nd International Conference on Signal Processing Systems (ICSPS)*, Dalian, China Vol. **1**, 2010; V1-599–V1-602.
18. Jia X, Shao J, Jing J, Liu P. CCA-secure type-based proxy re-encryption with invisible proxy. In *IEEE CIT 2010*, Bradford, West Yorkshire, UK, 2010; 1299–1305.
19. Hu J, Wang XA, Zhang M. Toward constant size CCA-secure multi-hop proxy re-encryption. In *2nd International Conference on Signal Processing Systems (ICSPS)*, Dalian, China 2010; V1-603–V1-605.
20. Schnorr C. Efficient identifications and signatures for smart cards. In *CRYPTO 1998, LNCS*, Vol. **435**. Springer: Heidelberg, 1998; 239–251.
21. ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory* 1985; **31**(4): 469 – 472.
22. Fujisaki E, Okamoto T. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO 1999, LNCS*, Vol. **1666**. Springer: Heidelberg, 1999; 537–554.
23. Bellare M, Boldyreva A, Desai A, Pointcheval D. Key-privacy in public-key encryption. In *Asiacrypt 2001, LNCS*, Vol. **2248**. Springer: Heidelberg, 2001; 566–582.
24. Cramer R, Shoup V. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO 1998, LNCS*, Vol. **1462**. Springer: Heidelberg, 1998; 13–25.
25. Shi E, Bethencourt J, Chan HTH, Song D, Perrig A. Multi-dimensional range query over encrypted data. In *S & P 2007*, Claremont Resort, Berkeley, CA, USA, 2007; 350–364.
26. Kiltz E. Chosen-ciphertext secure key-encapsulation based on gap hashed Diffie–Hellman. In *PKC 2007, LNCS*, Vol. **4450**. Springer: Heidelberg, 2007; 282–297.
27. Gentry C. Fully homomorphic encryption using ideal lattices. In *ACM STOC 2009*. ACM New York, NY, USA, 2009; 169–178.